ENVIRONMENT AWARE MULTI-FAILURE NETWORK RESTORATION TECHNIQUES USING FUZZY LOGIC

BY

MOHAMMED YAHYA AL-DARWBI

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER NETWORKS

JANUARY 2017

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **MOHAMMED YAHYA AL-DARWBI** under the direction of his thesis adviser and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN COMPUTER NETWORKS**.

Thesis Committee

Dr.Uthman Abdurrahman Baroudi
(Adviser)

Dr. Mohamed Younis (Member)

Dr. Marwan Abu-Amara (Member)

Dr. Ahmad Almulhem
Department Chairman

Dr. Salam A. Zummo
Dean of Graduate Studies

22/1/17
Date

عمادة الدراسات العليا
DEANSHIP OF GRADUATE STUDIES

To the greatest man in my life, my father may Allah make the paradise his destiny and forgive his sins.
To the most affectionate woman in my life, my mother may Allah grant her health, happiness and heaven .

ACKNOWLEDGMENTS

After my thanks to Allah, my first acknowledgment goes to my adviser Dr Uthman Baroudi for his continuous help and support. Without his guidance, I would not be able to get such achievement. I also wish to thank the other members of my thesis committee Dr. Mohamed Younis, and Dr. Marwan Abu-Amara for their constructive support and encouragement. My thanks and acknowledgments are given to KFUPM for providing me with all needed resources and facilities. Without such support, I would not reach this very important milestone in my life. Very special thanks to my beloved mother and father and to my dear wife, who remains willing to engage with the struggle, and ensuing discomfort, of having a partner who is busy with the studies, and research. My greatest gratitude goes to Mr. Gamal Sallam and Mr. Mohammed Al-shaboti for their help in achieving this work. Finally, I would like to thank all my friends Salah Alrabeei, Ibrahim Althamari, Al-Ezy, Yousef, Ahmed Al-Areeq and all my other friends for their support and encouragements and ideas - you guys were more than just friends.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

WSRN	Wireless Sensor and Robot Networks
WSN	Wireless Sensor Networks
CoRFL	Connectivity Restoration using Fuzzy Logic
NR	Networked Robots
RPP	Robot Path Planning
LPP	Local Path Planning
GPP	Global Path Planning
HPP	Hybrid Path Planning
D*	Dynamic A* path planning
PD*	Potential field D*
CoD	Center of Deployment

N_b	Number of neighbors
N_f	Number of Failed neighbors
U_{attr}	Potential field attractive force
U_{rep}	Potential field repulsive force
D	Distance
T_d	Travelled Distance

THESIS ABSTRACT

NAME: Mohammed Yahya Al-darwbi

TITLE OF STUDY: Environment aware multi-failure Network Restoration techniques using Fuzzy Logic

MAJOR FIELD: Computer network

DATE OF DEGREE: January 2017

Wireless Sensor and Robot Networks (WSRNs) are getting more attention in a wide range of industries. This type of networks can be deployed in harsh environments or where human intervention is risky or impossible. In these environments, WSRNs may suffer simultaneous failure of multiple nodes causing the network to be partitioned into disjoint parts which make these robot unable to communicate with each other and eventually unable to fulfil their mission. Existing approaches in the literature have mainly focused on restoring network connectivity while taking into account only the number of mobile nodes and the traveled distance cost and ignoring important factors such power-level of the robot, WSRN lifetime, and environmental conditions that constrain the robot motion. These conditions may hinder the robot travel on a direct path between two positions.

In this thesis, a distributed node-repositioning algorithm based on fuzzy logic (CoRFL) is proposed to reduce the number of exchanged messages for network failure detection, restore the network connectivity, enhance the network lifetime, and optimize the selection of the recovery path in a partially known environment. To minimize the intra-segment partitioning, CoRFL excludes critical nodes for network connectivity from participation in the recovery process. Two improved versions of CoRFL are also presented, namely CoRFLN and CoRFL2. In CoRFLN, if qualified a critical node is to participate in the recovery process, it gets replaced by its nearest node. On the other hand, in CoRFL2, fuzzy logic is employed again to find the best-replacement of relacotaed critical nodes.

Once, the recovery team is selected, the most suitable recovery path is to be determined. A direct travel path is not always possible; therefore, three different approaches are proposed to find the safest, shortest, and collision-free recovery path, respectively. PD* is the proposed algorithm for finding the collision-free path by exploiting the power of Artificial Potential Field in avoiding the obstacles to smoothen the path generated by a grid-based algorithm. The second algorithm is a Smoothed PD* which shortens the generated path by utilizing Dijkstra algorithm. The last one is Secure PD* which pre-prepares the secure level of passing through space and find a hazard-free path. Extensive simulation experiments have been conducted to validate the above algorithms. The simulation results demonstrate outstanding performance compared to sample competing approaches in terms of the total travel distance, average remaining energy of the recovery

team, and the lifetime of WSRNs. Moreover, a proof-of-concept real experiment using Khepera IV robots has been carried out for validating the proposed algorithms.

ملخص الرسالة

الأسم: **محمد يحيى الدروبي**

عنوان الرسالة : **إستعادة الاتصال في شبكات الروبوت باستخدام المنطق الضبابي مع الأخذ بعين الاعتبار التغيرات البيئية المحيطة.**

التخصص: **شبكات حاسوب.**

تاريخ التخرج: **كانون الثاني 2017**

إن التطور الكبير في تقنية الاتصالات وصناعة المعدات الدقيقة جعل تصميم وصناعة شبكات الاستشعار والروبوت اللاسلكية (WSRNs) حقيقة ماثلة للعيان، مما أكسبها الاهتمام على نطاق واسع في كثير من المجالات الصناعية والأمنية. فهذا النوع من الشبكات يمكنه الانتشار والقيام بعمليات البحث والإنقاذ في البيئات القاسية أو التي يحظر التدخل البشري فيها أو يستحيل. في مثل هذه البيئات، قد تعاني WSRNs الفشل في وقت واحد مما يسبب في تمزق شبكة الاتصال إلى أجزاء مفككة، تجعل هذه الروبوتات غير قادرة على التواصل مع بعضها البعض، وغير قادرة في نهاية المطاف على تحقيق المهمة الموكلة إليها. هناك العديد من الطرق المقترحة في هذا المجال ومع ذلك، فكل منها ترتكز أساسا على إعادة الاتصال بالشبكة آخذة بعين الاعتبار عدد العقد (robots) المتحركة وتكلفة المسافة المقطوعة فقط بينما تتجاهل العوامل الهامة الأخرى مثل: قدرة العقدة نفسها (الطاقة المتوفرة فيها)، عمر الشبكة بعد الحادث، والتغيرات الحادثة في البيئة والتضاريس المحيطة بها. هذه التغيرات في البيئة قد تبطل الافتراض المعمول به في سائر الأبحاث الحالية من حيث افتراض وجود مسار مباشر.

في هذه الأطروحة، يقترح الباحث خوارزمية (CoRFL) لإعادة تموضع العقد على أساس المنطق الضبابي (Fuzzy Logic) والذي بدورة يعتمد على عدة عوامل وهي: المسافة وكمية الطاقة عدد العقد المجاورة، وذلك للحد من عدد من الرسائل المتبادلة للكشف عن فشل الشبكة، واستعادة الاتصال بالشبكة، وتعزيز عمر الشبكة، وتحسين إختيار مسار إعادة الاتصال في بيئة معروفة جزئيا. فباستخدام CoRFL، فإن العقد (robots) التي تمثل نقاط ارتكاز حرجة في الشبكة (cut-vertex) لا تشارك في عملية إعادة الاتصال، وذلك للحد من انقسام الشبكة مرة أخرى. ومن ثم قام الباحث بإدخال إصدارات محسنة ل CoRFL وهي CoRFLN وCoRFL2. ففي CoRFLN، إذا تأهلت عقدة من نوع (cut-vertex) للمشاركة، فأقرب العقد اليها يحل محلها. ولكن عند تطبيق CoRFL2، فإنه المنطق الضبابي يعمل مرة أخرى للعثور على أفضل عقد لاستبدال عقدة (cut-vertex).

وبعد تحديد فريق إعادة الاتصال، نبدأ البحث عن المسار الأنسب، فالمسار المباشر قد لا يتوافر دائما. ولذا اقترح المؤلف ثلاث طرق لحل ثلاث مشاكل رئيسية وهي العثور على مسار قصير، أمن، وخالي من التصادمات لكي يصبح الروبوت قادراً على الوصول إلى هدفه في بيئة متعرجة ومتغيرة وعرضة للتدمير. الطريقة الاولى لحل مشكلة التصادمات مع العوائق المتواجدة في طريق الروبوت هي *PD، وهي تعتمد على مبدأ التجاذب والتنافر بين الروبوتات والعوائق والمستخدمة في خوارزمية APF لتحسن المسار الناتج عن خوارزمية *D. الطريقة الثانية لإيجاد اقصر مسار ممكن هي *Smoothed PD، وهي تقوم بتقصير المسار باستخدامها الخوارزمية المشهورة Dijkstra. الطريقة الاخيرة هي *Secure PD، والتي تقوم بتقييم مستوى الخطورة لأجزاء المنطقة التي سوف يمشي عليها الفريق المكلف بإعادة الاتصال ثم تطبق الطريقة السابقة *PD.

وقد أجريت تجارب محاكاة واسعة النطاق للتحقق من صحة الخوارزميات المذكورة أعلاه. وتبين نتائج المحاكاة الأداء المتميزللطرق المقترحة بالمقارنة مع الأساليب التقليدية من حيث كمية الطاقة المتوفرة لدى فريق الاسترجاع، إجمالي مسافة السفر وزيادة عمرWSRNsعلاوة على ذلك، تم إجراء تجربة إثبات الصحه باستخدام عشرة روبوتات Khepera-IV للتحقق من صحة الخوارزميات المقترحة.

CHAPTER 1

INTRODUCTION

In this chapter, we will give an introduction to: robots , Robotics Networking (RN) and its underlying technologies ,Dead Reckoning Localization , and Fuzzy Logic which is the optimization tool used in the selection process.

1.1 Robots

Advances in technologies like accurate sensing devices , efficient and powerful robots motors, and communication technologies have enabled the development of low-power robots which have the capabilities of sensing, processing, and communicating. A robot is usually equipped with sensors to probe its surrounding environment such as distance measurement and temperature sensors. The most prominent robotics applications are those where human intervention is limited or denied such as search and rescue operations, security surveillance etc. Robots could be also used for applications where there are economic benefits for using mobile sensors such as farming or production line applications. Robots are per-

forming tasks that require them to act on information derived from sensors, co-operate with humans, and coordinate with other robots [26]. Robots applications like searching, exploring, and mapping, are performed faster if the number of robots is large. Therefore, deploying multiple robots carrying out tasks in parallel is required. Robots deployment in an area of interest can be done either manually by placing each robot in a pre-determined location , or robots may dynamically adjust their positions by self-spreading.

1.1.1 Networked Robots

Networked Robots(NR) refers to multiple robots, a collection of robots that are resident at identified locations on a network , operating together in coordination or cooperatively with deployed sensors, human users , and centralized computers [26]. From the networking point of view, a robot constitutes a single entity and the robot actuation capabilities and mobility advantages are utilised efficiently when it becomes a part of a robots network [26]. In addition, NR allow robots to perform the required tasks that are greater than the abilities of a single robot. The main challenge of using NR is the robustness of multi-robot network against failure and parallelism operation. In NR , robots can be used to deploy a self-organized network whose functions are to sense the surroundings in order to detect a certain condition or event, process the data, send information to a management station , execute tasks and control an actuator. Therefore, the choice of robots is obviously controlled by the application requirements.

Wireless Sensor and Robot Network (WSRN)

The uses of robots as a mobile actor within sensor network, the terms Wireless Sensor and Robot Network (WSRN) is used [7, 51]. Wireless Sensor Network (WSNs) are widely used in various areas of our life such as industrial application, environment monitoring, and military. Obviously, robots are equipped with more energy and processing capability and they can communicate through longer distances, unlike sensors. When a sensor detects an event, it first notifies a nearby robot which analyzes the data and coordinates with other robots to take the required action. For example, in a forest fire detection application, sensors detect fire and send notifications to the robots where they can coordinate among themselves and extinguish the fire before spreading to the rest of the forest.

WSRN is typically deployed in harsh environments like a disaster or a remote area or even a planet such as Mars to accomplish a specific mission. Such environments put the network under several constraints namely, sudden wide damage, difficult human intervention, etc. Therefore, a pressing network design issue is to design a self-healing network that can restore network connectivity after a sudden wide damage that may cause the deployed network to be partitioned into many completely isolated parts . Two main factors put the connectivity of WSRN at risk.

- First, sensors and robots are constrained by energy supply , and bandwidth capacity.
- Second , WSRN is typically deployed in harsh environments to achieve a

specific mission. If a sudden wide damage cause the deployed network to be partitioned into many parts , a self-healing network is required.

Therefore, most researchers have worked to address these requirements when they designed their protocols and algorithms. In order for a WSRN to carry out its application successfully, it has to satisfy the following requirements:

Localization: Due to the movement ability of the robots , the exact location of the robot is important. Without knowing the location, deployment and task allocation would be impossible. Accurate robot measurement devices and a precise localization method are required.

Path Planning: Along with the localization algorithm, path planning is crucial in WSRN because a good path planning algorithm increases the network lifetime by minimizing the robots wasted power.

Coordination: Unlike WSN where there is a single entity (i.e. the sink) which receives all sensed information and deliver it to a central monitoring system, WSRN needs a coordination mechanism among robots to carry out the required task.

Real time : robots have to propagate the data to each other in real time and robots have to act promptly. For example, in fire detection application, any delay renders the late action useless or impossible.

Self-healing: It describes the network that has the ability to perceive that it is not operating correctly and, without human intervention, make the necessary adjustments to restore itself to normal operation.

The above requirements impose difficult challenges in the design and development of a sustainable WSRN . WSRNs are deployed usually in a hostile and harsh environment thus they are prone to frequent failures that could render the network useless if there is no self-mitigation to such failures. Moreover, these nodes are battery operated and therefore their energy might be exhausted at any time.

1.2 Path planning

The essence of the path planning problem is to find the shortest path from the robot position to a defined target position. The optimized path from the start to the end location is the output while the input is the environmental factors. Due to several technological advances, path planning has been extensively used in different fields not only in robotics but also in games, manufacturing, automotive applications, aerospace applications, etc [19]. In Robot Path Planning (RPP) , a robot and a description of an environment are given. The goal is to plan a path between two specific locations such that the path must be collision-free and satisfy certain optimization criteria. RPP is considered one of the main issues in the research area of robot. The descent RPP algorithm should find a collision-free path through a known, unknown or partially known environment as well as to optimize it with respect to some criteria. According to [50], RPP classification is divided into Local Path Planning(LPP) based on sensor information which is done in a completely unknown environment , Global Path Planning (GPP)based on a

completely known environment (static environment and obstacles) called Pianos Mover, and Hybrid Path Planning(HPP) based on partially known environment.

Path planning algorithms that are used in an unknown environment or dynamic environment are called (**LPP**), Real-time Path Planning (RPP) or On-line Path Planning (OPP). Finding a path in an unknown environment with obstacle avoidance is considered as an NP-Hard problem [19]. LLP algorithms first carry real-time navigation and plan only for the next move for the robot to follow in order to reach the target location. LPP can be defined as a real-time or on-line path planning with obstacles avoidance. The **GPP** is not as hard as LPP because it is based on a completely known and static environment so no arbitrary moves are needed. One of the oldest algorithm of finding the shortest path between nodes in a completely know environments is a greedy algorithm called Dijkstra's (named after its originator, E.W. Dijkstra) . Finally, **HPP** techniques are a combination of online and offline techniques and . It first generates the path in an offline mode using the available information about the environment then switches to online mode if changes in the environment occur. The most popular algorithm for this purpose is D* (dynamic A*) proposed in [45] by Stentz, which solves A* algorithm problem in dealing with unknown or partially known environment to re-plan if necessary. D* incrementally repairs the path as a change in the environment is detected.

1.2.1 Path Planning in network recovery

In the last few years, there has been a growing interest in federating the partitioned network. The [35, 53, 54] references are the latest surveys on WSRNs partitioning recovery that present a complete knowledge about the problem and the existing solutions. It is generally thought that every node can participate in the recovery process. Despite the lack of power or the large node rank, they enforce the node to participate in the recovery. Finding a short and collision-free path to fulfill the connectivity restoration process needs a good path planning algorithm. The length of the path determines how many nodes are required to fulfill the connectivity restoration, and it is influence by the changes in terrain, environment, and obstacles. Because the network is partitioned as a result of unknown factors that affect the previously known environment, hybrid path planning techniques are the most suitable for solving such problem. It first generates the path using the previous information about the deployment area then it responds to the newly detected changes in the environment by finding a new suitable path.

1.3 Problem Statement

The existing approaches of connectivity restoration rely on three assumptions that make their implementation limited to very special environments. First, previous researchers assumed that the nearest node to the center of deployment should move to restore the connectivity without considering the power cost. Second, most existing works do not consider whether the moving node is a cut-vertex or

not. This blind selection of the recovery team may minimize the network lifetime, maximize the total travel distance, and disrupt the network within the partition. The third assumption is that a direct path from the starting point to the target location is always possible. Sometimes the direct path is not possible due to the dynamic changes of the environment. Not only this, but also a completely known environment is not always feasible because a failure might happen due to changes in the environment. Such assumptions are inconceivable where the presence of obstacles and the possibility of environment changes are highly expected

1.4 Thesis contribution

In this thesis we tackle the above mentioned problems. The objective is to develop a recovery mechanism that takes into account the power level of the node and its rank and not only the distance. Furthermore, the path of the recovery team should be short and collision free. The primary contributions of this work are:

- **Detection broadcast messages overhead**: The number of messages that is required to identify the partition size, and inform the unaware nodes is minimized.

- **Recovery team selection** : we exploit the power of Fuzzy logic to evaluate the capability of each node and decide whether it is liable to participate in the recovery process. Three main metrics are the input of the fuzzy logic system. These metrics are:

- **Travel distance**:The robot with minimum required travel distance is more suitable to become one of the recovery team.
- **Level of power** : The robot with high power is preferable to become one of the recovery team.
- **Node rank**: The higher rank the robot has, the more probability to include it in the recovery team.

Based on the aforementioned metrics, the recovery team are chosen carefully to minimize the total travel distance as well as the possible future failure and to maximize the network lifetime. In search for incredible, four approaches are proposed:

- **Most qualified** : the most qualified nodes are chosen to form the recovery team.
- **CoRFL**: The most qualified nodes are chosen as a recovery team but the cut-vertex nodes are excluded.
- **CoRFLN**: If one the most qualified nodes that have been chosen to be recovery team is a cut-vertex and its movement will split the network into more partitions, its nearest node will replace its location to give it the ability to move without disrupting the network.
- **CoRFL2**: The recovery team are those who are most qualified nodes and if one of them is a cut-vertex and its movement will split the network into more partitions, the node(s) with high power and least

travel distance will be nominated for the replacement task.

- **Path Planning**: Since the straight recovery path is not always possible, three dynamic path planning approaches are proposed. The first one is to find shortest path possible, the second one is to make the shortest path collision-free, and the third one is trying to make the recovery path as safe as possible by converting the dangerous known or predicted area into non-traversable areas.

1.5 Organization Of Thesis

The rest of this thesis is organized as follows. Chapter 2 surveys the related work for both network restoration and path planning. Chapter 3 covers our new WSRN connectivity restoration approaches. In chapter 4, novel connectivity restoration path planing algorithms are presented.

CHAPTER 2

RELATED WORK

Indeed, all the restoring connectivity techniques in the literature have not focused on both the recovery process and path planning altogether. So the work in the literature will be classified into either a recovery process or path planning in a partially known environments.

2.1 Network Restoration

A huge amount of works about WSRNs connectivity restoration issues have been proposed. In terms of how reestablishing the connectivity is achieved those researches do it either by deploying additional relay nodes or by repositioning healthy nodes from the partitions.

Deploying additional nodes has been studied extensively in the literature and most of them are centralized. Those additional node can be stationary, mobile or mixed.in [20, 37, 38, 40, 41], a stationary nodes are proposed which act as a relays between segments without movement, and they are thrown manually ether by a

plane or rocket. In [4, 5, 39], a mobile nodes are used to restore the connectivity with movement which minimize the number of deployed nodes. In [1, 43, 44], a mix of mobile and stationary relays are proposed. The partitioned network in a harsh environment cannot be solved using a centralized approach because most of the separated partitions will lose the connection to the central point. In addition, deploying new relay nodes is not applicable in such environment for example if the network is deployed indoor or under water, it is sometimes impossible to through new nodes to recover the connectivity. For this purpose, those types of solutions will not be included in this work. Only the repositioning of the healthy nodes approaches will be surveyed.

Node repositioning approaches based on partial or uncertain information about the number of failed nodes and locations of the healthy nodes. Obviously, a lot of work is done to restore the collected node failure by repositioning a few healthy nodes, but it is still very challenging. The main challenges are determining the damaged area and minimizing the future failures. The issue of WSAN post lifetime after connectivity restoration has not been considered well in the existing literature. Even more, most of the literature work did not focus on the energy of the healthy nodes, did not suggest an intelligent scheme to solve the problem and did not even mention the importance of path planning. In [37], Distributed Optimized Relay Node Placement Using Minimum Steiner trees (DORMS) is a distributed algorithm proposed to federate partitioned network and it assumes the center of deployment (CoD) is known in advance. Every partition deploys

movable nodes towards the CoD. The closer nodes to the damaged area will be a moving node, and the leader is the one with the highest number of lost connections with the neighbors. In addition, work in [37] did not tackle the expected huge message overhead due to the rush of failure neighboring nodes to propagate this event. In [20, 40], the same authors proposed a distributed algorithm to recover multiple node failures and it assumes the sink node location is well known for the whole network sensors. The proposed algorithm based on the pre-failure routing information in order to determine the location of the damage. Sink node is responsible for collecting the sensing information from the sensor. The failure is detected if there is no connection with the sink node. When the path from every node to the sink node is established, the location of path nodes is collected. Thus, on a partitioning happened, the node can reconnect to the sink node by moving to the next hop. A cascaded movement is used to reconnect k-hop neighbors with the sink node. The recovery leader governs the cascade movement; every k-hop neighbors should agree on one leader. This leader is the closest node to the sink. This means that every partition is divided into many small parts and every part is k-hop neighbors.

As we have seen, the authors did not consider the power consumption in the recovery process when they proposed constructing many recovery teams that surely will increase the overall power consumption and consequently shortens the network lifetime. Moreover, this work did not minimize the broadcast overhead that will be generated when the failure event is detected. In[38], AuR is a distributed

algorithm and it assumes the CoD is known in advance. It is based on a block movement unlike the others cascade movement; therefore all nodes are involved in the recovery. As a group, they will spread outward to extend their area trying to find a connection with the failed nodes if there is no connection they will move together toward the CoD. Once they reach the CoD, they will spread to maximize the coverage area.

Meanwhile in terms of environment reality, only [5] is proposed a restoring connectivity approach that takes into account the terrain-aware and obstacles. In [5], not a direct path is assumed, but a static environment is assumed which are not always possible. It is an advanced version of their algorithm Route-Based Approach (RBA) where the restoring network connectivity is based on restructuring the network topology while minimizing the energy cost due to the movement. It enhances the leader selection and the cascade movement. A* [10] algorithm, which is widely used to find the least-cost path from a given initial location to the target location, is used to plan the recovery path in the assumed completely known environment. In [3], a virtual tree is built for coordinating the recovery process among nodes in the partitioned area. In this tree, each node is assigned a recovery weight as well as a nearby cluster member, which serves as a gateway to lost nodes. Based on the number of children associated with a specific node in the recovery tree, this node is either designated as a cluster head or not.

.

2.2 Path Planning

With the technology advances, path planning has been extensively used in different fields not only in robotics but also in games, manufacturing, automotive applications, aerospace applications, etc. Robot Path Planning (RPP) has been considered as one of the main issues in the research area of the autonomous mobile robot. To plan a path between two specific locations using a given description of an environment which the robot walks on, the path must be collision-free and satisfy certain optimization criteria. A complete RPP algorithm must satisfy two criteria:

1. It must always find a solution if one exists, in a finite time.

2. If no solution exists, it must report this, in a finite time.

In this section, a brief literature review about both Local and Hybrid path planning will be presented and global path planning is excluded because it is out of our scope.

2.2.1 Local Path Planning

Path planning algorithms with obstacles avoidance that are used in unknown environment or dynamic environment are called Local Path Planning (LPP), Real-time Path Planning (RPP) or On-line Path Planning (OPP) . As finding path in unknown environment with obstacle avoidance is an NP-Hard problem, so LLP algorithms obtain the path gradually by planning only for the next move which

the robot has to follow in order to reach the target location. LPP algorithms classified into three main areas: Rapidly-exploring random (RRT)[18];Bug [24]; and Artificial Potential Fields [14].

RRT is a probabilistic technique proposed in [18] to generate a path with based on local environment constraints; in response to the detected obstacles or environment changes. The idea behind RRT is that it imitates the growing of tree rooted at the starting location, only the collision-free tree branches grow up, by using random samples from the search space. RRT tree is grown by moving toward the unexplored areas in the environment under search, and feasible paths only are added to the tree. RRT tree stops growing when the target location is reached by a certain tree branch, and the final path is obtained by tracing back from the target location to the start location. Both environment discovery and path planning are done simultaneously. The main drawbacks of RRT are its random growing in many directions which it is not viable in robotics because of its time and energy consuming.

Bug algorithms[24], a well-known sensor-based robot navigation and path planning technique; the basic idea is to follow the initial path which is a straight line between the start and the target locations called main line (M-line), and to turn around the detected obstacles by following the perimeter of the obstacle. Bug obstacle avoidance will circumnavigate the detected obstacle. Bug identifies the location using the sensing data during the robot movement to estimate the change in the location over time. Mapping or modeling the environment in Bug family

algorithms is not required and only the sensor sensing information is required. In order to achieve a better performance, shorter path, minimum execution time, and simpler algorithm many variations of Bug algorithms are proposed in the literature.

Artificial Potential Field (APF) method is one of the well-known local path planning that was first introduced by Oussama Khatib in [14] and it is also called Virtual Force method. The potential function can be defined over free space as the sum of an attractive potential pulling the robot toward the goal configuration, and a repulsive potential pushing the robot away from the obstacles. With the potential field algorithms, the environment is modeled to generate by assigning the target with an attractive force and the obstacles with a repulsive force. In an iterative fashion, motion planning is performed. At each iteration, the artificial force induced by the potential function at the current configuration is regarded as the most promising direction of motion, and path generation proceeds along this direction by some increment. The generated path using APF is based on influencing the robot using the sum of two forces, attractive and repulsive potential forces, where the robot is pulled toward the target location using the attractive force and the collision between the robot and obstacles are avoided by pushing the robot away using the repulsive force. APF is known for its elegant mathematical equations and simplicity in finding the path with very little computation [27]. Artificial potential field method has the lowest energy at the location of the goal and can work without full map information. In addition, the advantages of

obstacle avoidance algorithm based on the artificial potential field has low computational complexity and good real-time characteristics [27]. But APF has its own shortcomings which are:

- the robot is stagnant or trapped in the presence of Local Minima (LM),
- only one path is generated but not the shortest path.

APF a major problem is LM, which can trap the robot before reaching its goal. LM is happened where the attractive and repulsive forces are equal, so robot makes no progress which is a dead end. The artificial potential field method bottleneck lies in its inability to counter this issue of local minima in the working environment. This situation leads to the robot getting stuck in a position and being not able to compute its next move [6]. In the literature there are many proposed solutions to overcome the problem of local minima like:

- Complementing LM with influential algorithms to escape from trap [17],
- Using escape-force [49],
- Introducing a trap recovery model [25],
- Proposing an adaptive virtual target algorithm [23]
- Overcoming by using global planner [13]
- Replacing LM with global minima [2]
- Exploiting harmonic functions as a solution to Laplaces equations [33].

2.2.2 Hybrid path planning

Both global and local path planning algorithm trap when a great change in the environment happened. Hybrid path planning algorithm solve the shortcomings of both local and global path planning. D* is the most popular HPP algorithm . D* is an improved version of popular global path planning algorithm which called A*. As a change in the environment is detected, D* incrementally repairs the path .

It represent the area as a grid as shown in figure(2.1). As figure (2.2) shows, the cost of the movement from one cell to another is varied. Figure(2.3) shows that, D* start to compute the movement cost from the goal until it reach the start point. The obtained path is followed unless a new obstacle is detected. Once an obstacle emerges , D* modifies the path.

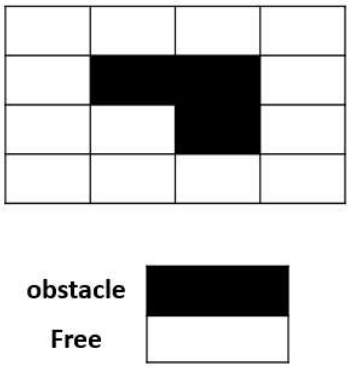


Figure 2.1: D* area representation.

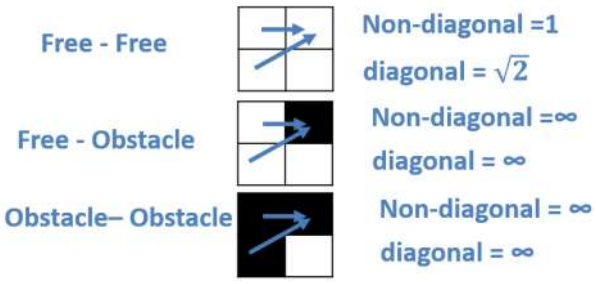


Figure 2.2: D* movement cost from cell to another.

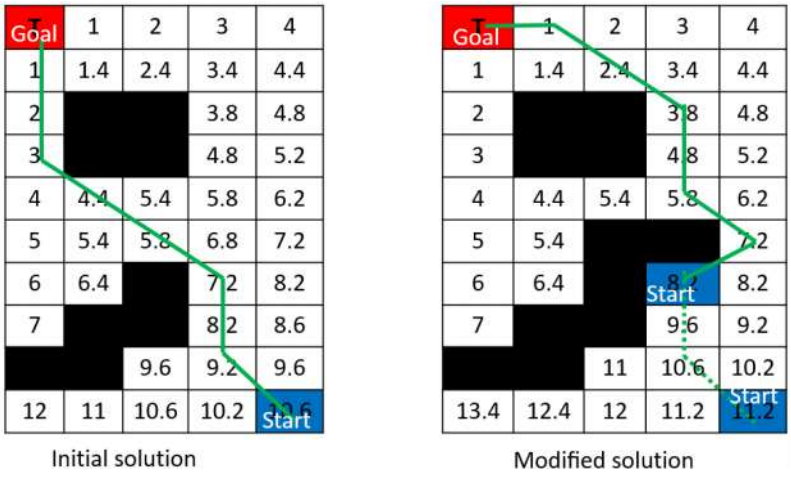


Figure 2.3: D* path example .

Many D* variants are proposed in literature to improve its execution time. In [46] Focused D* proposed by the same author to improve the computation cost of the original D* and to optimize the subsequent operations. In [16] Lifelong Planning A* is proposed which is an incremental version of A* also called continual planning. Lifelong Planning A* (LPA) name comes from its nature of reusing the information from the previous searches. A* is used in the first run of LPA where all nodes are involved, but in the subsequent search number of involved nodes is reduced in order to make the search faster by reusing the previous search. Unlike

D*, it continuously determines the minimum cost path from the initial location to the target location.

In [15] D* Lite is similar to D* in the navigation strategy except algorithmically they are deferent. D* Lite is a result of applying Lifelong Planning A* to robot navigation in unknown terrain and not to mention its simplicity. Like D* it finds the shortest path from current location to the target. In [28] DD* Lite is an efficient incremental search algorithm for issues that can capitalize on state dominance. It uses D* Lite along with dominance relationships technique detection to solve robot navigation with global constraints.

In [34] E* is a generic path planning method that combines path cost interpolation with dynamic replanting capabilities which lead to significant fidelity navigation. E* is inspired from D* and its variants to make the path planning flexible for the dynamic environment which is also based on Level Set Methods(a conceptual framework for the numerical analysis of surfaces and shapes using level sets as a tool). In spite of using the grid-based environment model, it is free from grid-based calculations discretization effect.

In [12] ADAPTIVE A* is an incremental heuristic version of A* that solves series of similar search issues in a fast way. Its contribution is to use the previous searches information to update the h-values (heuristic parameters) of A*.According to [47] Adaptive A* is simple to understand and easy to implement because it basically transforms consistent h-values with respect to the goal state into more informed consistent h-values with respect to the goal state, which allow

it to get the shortest path where the action costs can increase with time. Adaptive A* authors claim that it is 10 percent faster than A*. In [22] Generalized LPA* (GLPA*) is a framework that generalized LPA* which gives LPA* ability to be developed to many capable versions. According to the given optimization criteria, gives one a considerable amount of freedom, GLPA* always find the best path from the given start location to the target. In [47], a Generalized Adaptive A* (GAA*) is presented. The Generated path using Adaptive A* is not guaranteed to be the shortest where the action costs decrease with time because after action cost decreases, consistent h-values do not remain consistent. GAA* solve this issue by correcting h-values after action cost decreases.

In [52] Field D* is an extension for D* and D* lite which produce a globally-smoothed paths using the linear interpolation. Unlike the previous algorithms, Field D* is based on any-angle movement not grid-based. Theta* introduced in [30], Both Theta* and Field D* are variants of A* and are based on any-angle movement. It gives a shorter path than Field D* by propagating information along grid edges without constraining the paths to grid edges. In [31] Incremental Phi* is proposed which combines the incremental strategy (D* Lite), reuse information from the previous search to speed up the next one, with Theta* which is any-angle movement. In [32] Lazy Theta* is proposed which is an extension for Theta* to speed it up in cubic grids. It reduces the number of collision checks (line-of-sight) by delaying them till they are necessary.

Dynamic Fringe-Saving A*(FSA*) is presented in [48] which is an incremental

version of A* that gets the shortest path repeatedly from fixed location to a fixed target location in a known environment in case of changes in cells traversability overtime. In FSA* the beginning of the previous A* search that is identical to the current A* search is reused. M* is proposed in [14]. As the number of robots grows exponentially, multi-robot path planning becomes difficult. Multirobot path planning is difficult because the configuration space of the system grows up according to the number of robots.

CHAPTER 3

NETWORK RECOVERY USING FUZZY LOGIC

Our proposed recovery strategy is based on fuzzy logic in selecting the recovery team. Fuzzy logic is a well-known model with a powerful theoretical background that combines approximate values of any range. Because of the uncertainty in choosing the most capable nodes to restore the connectivity, the powerful capability of fuzzy logic is used to handle it. The recovery process is mainly divided into three main parts .The first part is the failure **detection** which is responsible for determining where and when the failure occur and try to minimize the broadcasting messages overhead. One of the detecting nodes will act as the partition supervisor of the recovery process. Supervisor is the one with the most dead neighbors, and if more than one have the same dead neighbors then the one with higher power is the supervisor. The second part is the **recovery team selection**, which is executed as a result of the detected failure . The third part is the recovery

path planning which is presented in the following chapter.

During the deployment process the Center of Deployment (CoD), the most viable point that is near to center of mass , is identified and its information is sent to all nodes. As it can be seen in the scenario presented in figure(3.1) , after a massive damage the network gets partitioned into four completely isolated partitions which is a critical problem because most of the network functions could not be achieved.

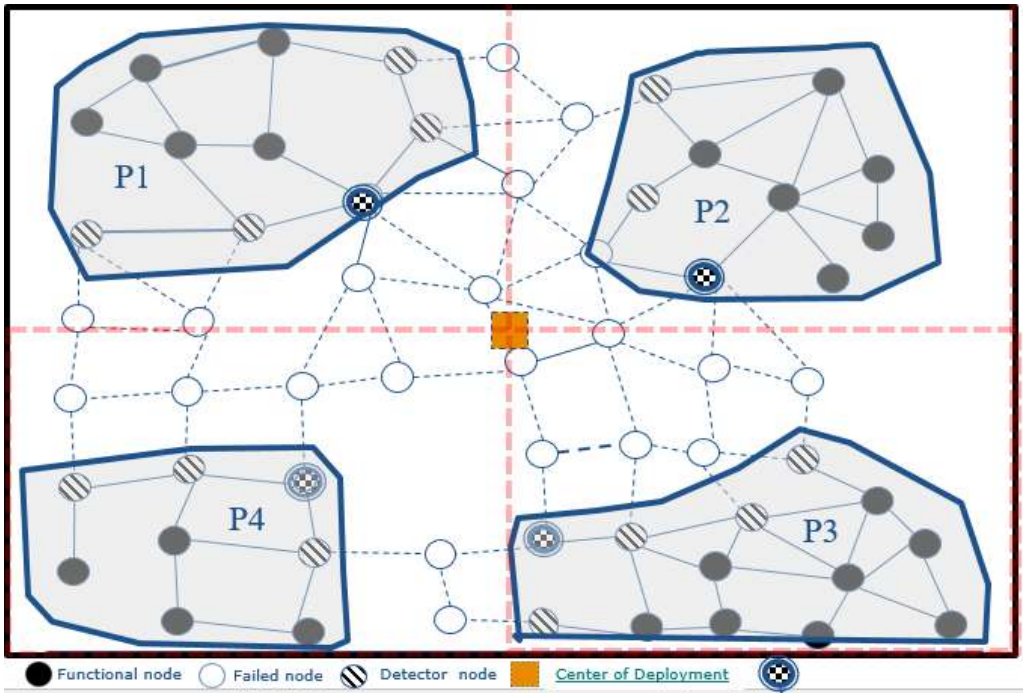


Figure 3.1: Network failure scenario.

Our proposed algorithm solves the partitioning problem through three phases. The first phase governs the normal operation before any failure. The second phase is the failure detection which is responsible for determining where and when the failure. The last phase is the recovery phase, which is executed after the failure

event. The recovery phase is divided into two main parts, recovery team selection and recovery team path planning where the later is presented in the following chapter.

3.1 Normal Phase

Once the network setup is completed successfully, every node has to maintain its two-hop neighbors table by sending a broadcast heartbeat message with TTL=2 to introduce itself to its neighbors. This neighbors information table is updated periodically. This table contains the important information about the neighbors like NodeID, NodeRank, Nodelocation. NodeID is a unique identification number, which is assigned to the node at the network setup. Noderank is the number of neighbors of this node. Nodelocation is the Cartesian coordinates of the node (xi,yi). In addition, we assume that each node can determine whether it is a cut-vertex or not.

Broadcast algorithms for wireless ad hoc have been proposed in the literature such as Gossip3 in [9] and PCBR in [29]. In case of network restoration, no specific algorithm is presented. To minimize the overhead generated by sending a broadcast heartbeat message every (t) seconds PCBR algorithm is used with some amendments. PCBR algorithm is presented in figure (3.2). PCBR retransmissions is always delayed by RAD timer **(every t seconds it is triggered).** Where h is the threshold value of the number of copies and p is the the threshold of the transmission which is based on the number of neighbors, as show in equation (3.1).

The more neighbors the node has , the less probability it has to send a broadcast message. The node with less neighbors is at risk so it should keep the others aware about its status.

$$\mathbf{p} = \frac{1}{n_b} \tag{3.1}$$

```
function PCBR (n_{p,h})
    start a new message
    p = 1/ n_p
     repeat
        if broadcast message received Then
            m-> m+1
        end if
    until RAD expire
    If M<h then
        if The Generate Random Number <=P then
            Forward Packet
        else
            Discard Packet
        end if
     else
        Discard Packet
    end if
end function
```

Figure 3.2: PCBR Broadcast scheme.

3.2 Failure Detection

As it can be seen in the scenario presented in figure(3.1) ,when such massive damage occurs to the network, the proposed algorithm will detect the partitioning event when one of the following incidents happens :

- The closest nodes to the damaged area will detect the failure when there is no reply for periodically sent hello messages of maintaining the two-hop neighbor's information, or
- When the node becomes unable to any other node,
- Dropped messages have increased dramatically.

Every node detects the failure it will send a broadcast warning message to inform other nodes. This way shall cause huge flooding in the network. All what the nodes need is a single failure report. Afterward, each informed node will have to respond. If that node does not reply then there is a problem with that node. By doing so the complete scope of the failure will be determined. If no such mechanism is utilized and many node discover the failure they will overwhelm the network while a single notification is adequate.

In order to minimize the broadcast overhead, we propose a new broadcasting method , presented in Fig 3.3, that is used after the failure is detected . Whenever they detect the failure, each node computes its own threshold (h) based on equation (3.2) , and then it generates a random number between [0-1]; if it is less than its threshold, it will send; otherwise, it has to wait for a period before sending the broadcast message.

$$\mathbf{h} = \frac{n_f}{n_b} \tag{3.2}$$

where n_f is the number of failed neighbors, and n_b is the total neighbors of the detecting node before the failure event. If the waiting time of the node expires and no message is received reporting the problem, this specific node should send with probability equal one. This detecting node will act as the partition supervisor of the recovery process. The broadcast message contains the following, a failure warning code, (h) value , and power level of the sender. Once a failure detection broadcast message is received the receiver should wait for a short time to see if another messages arrive from others nodes. If more than one supervisor send a broadcast message, the node will send a single reply to the one who has the highest h. The receiver will reply once , If another copy from the supervisor received it will be discarded.

```
function DETECTE(np , nf)
    h ← nf/nb
    Generate Random Number P
   If P<h then
        Send a Broadcast
   else
        repeat
             if broadcast message received Then
                  m->m+1
             end if
         until time expire
         if m==0 then
             Send a Broadcast
         end if
    end if
end function
```

Figure 3.3: Broadcast scheme.

3.2.1 Broadcasting Simulation Setup and Results

Message overhead is one of the main issues in WSANs connectivity restoration. In the literature, most of the recovery algorithms assume that the nodes detecting failure will immediately send broadcast messages and flood the network. The same as in CoRFL, we reduce the number of messages exchanged by giving the highest priority to the node that lost most of its neighbors and it will initiate the failure announcement.

Simulation setup:

- 100 random topology is generated(each topology has at least four partitions). Figure(3.4) shows one of the random topologies used in the simulation. The nodes in black are the detecting nodes of each partition.

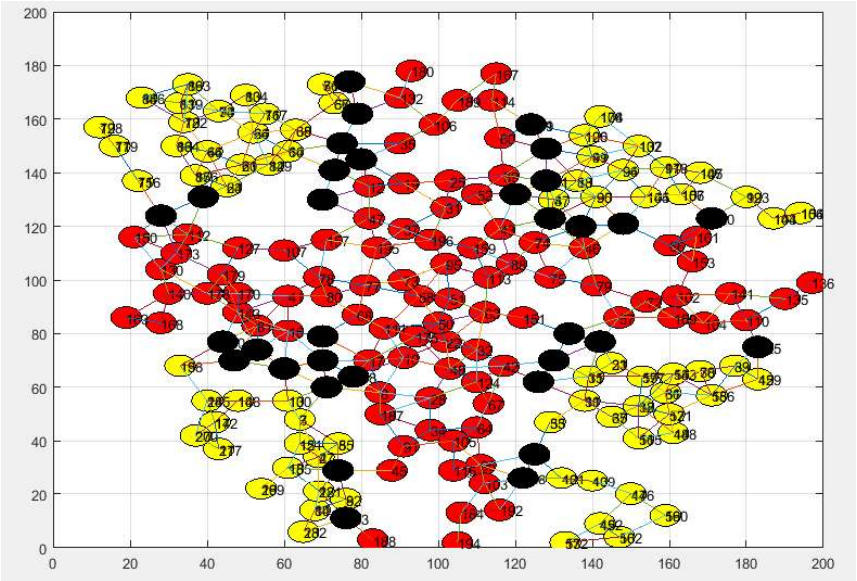


Figure 3.4: Broadcast simulation scenario example..

– (n) The number of detecting nodes is calculated for each partition

* (h) value of the algorithm is computed $h = n_f/n_p$ foe each detecting node.

* (p) value is generated randomly (10 times)

  · for each (p) , compare it to (h) and count how many nodes will participate.

* find the average of the participating nodes.

– averaging the participating nodes for the literature and proposed.

– theoretical messages overhead is computed.

Results: The following figures shows the results of the simulation.

1. The number of nodes that send a broadcast messages while the partition size vary as shown in figure(3.5).

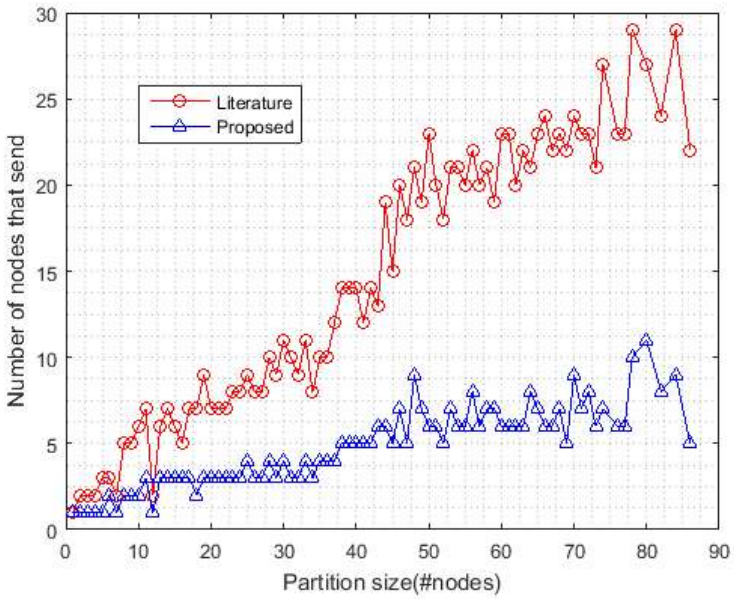


Figure 3.5: Network partition size compared to the number of sending nodes.

2. The number of sending nodes compared to the number of detecting nodes vary is presented in figure(3.6). In literature works every detecting node is a sending one.

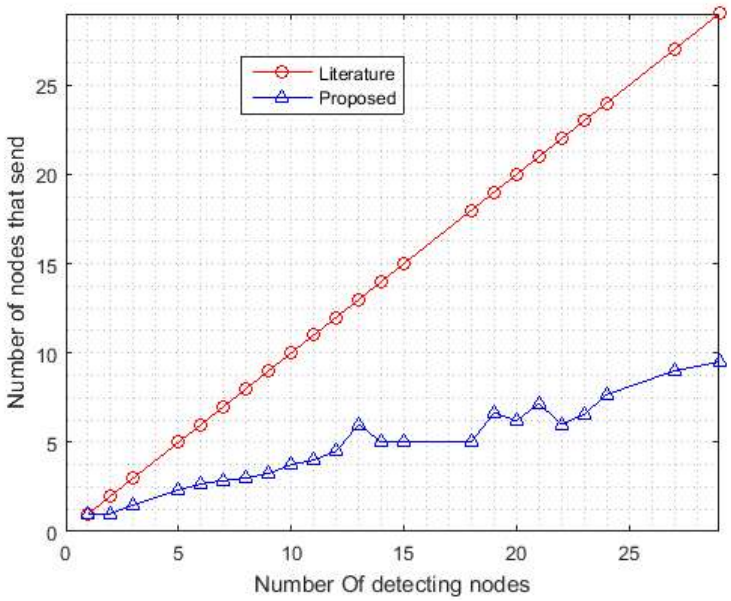


Figure 3.6: sending nodes compared to the number of detecting nodes.

Number of Messages Sent to Establish Recovery Process: As illustrated in figure (3.7), as the number of nodes increases, the message overhead grows dramatically in the literature recovery algorithms and DORMS is an example of them. This happens since all failure detecting nodes will send broadcast messages. On the other hand, in our approach, a few of the detecting node will send that makes CoRFL outperforms others. The optimal solution where only a single node sends a broadcast message is compared with the proposed and the literature to point out the goodness of our work.

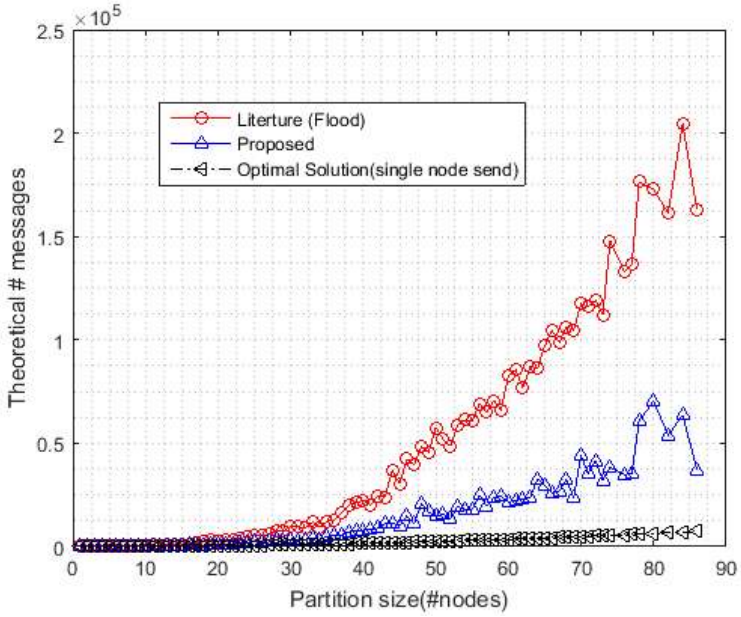


Figure 3.7: Messages overhead.

3.3 Information Gathering

One of the detecting nodes will act as the partition supervisor of the recovery process. Actually, supervisor is the one with the most dead neighbors, and if more than one have the same number of dead neighbors then the one with higher power is the supervisor. Every informed node should reply to the supervisor based on the received h values along with the broadcast message. Figure (3.8) shows the content of the reply message that is sent to the supervisor.

....	id	location	energy

Figure 3.8: Reply message content.

The partition supervisor starts collecting the information of the partition nodes, before the recovery process begins. It has to do the following:

- Wait for a specific time until the partition nodes information is received.
- Draw the initial recovery path that the recovery team should use to reach the CoD. A single path is mandatory in which the recovery team move one after another toward the CoD while they keep connected with the partition, figure(3.9.a). If each node move in its own shortest path, as in figure(3.9.b) ,the connection between them is not guaranteed which makes the network restoration harder.

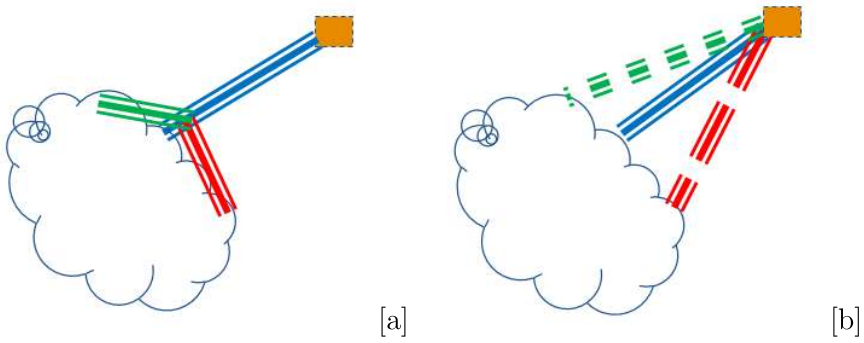


Figure 3.9: Single path verses multi path.

The closest node location to the CoD is the start point of the recovery path and we call it the pivot point. Figure(3.10) shows that the node in red is the pivot point of that partition.

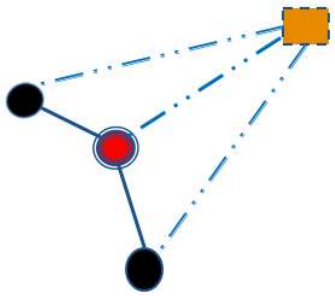


Figure 3.10: Pivot point identifying.

- Compute the node required distance from its location to the CoD. We define node distance as the summation of two distance components. As it is shown in figure (3.11), the first one is the distance from the pivot point to 'the CoD. The second is the distance from the node to the pivot point. The partition supervisor computes the node distance to CoD.

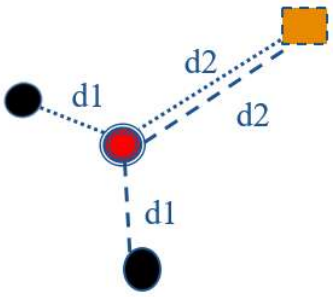


Figure 3.11: Node distance to CoD.

After identifying the recovery path, the process of selecting a recovery leader and its team will start. Fuzzy logic is used in the selection process.

3.4 FUZZY LOGIC SELECTION SYSTEM

Fuzzy logic [36] is used to handle the uncertainty on making the decision. It is a well-known powerful tool for dealing with approximate rather than exact values. The fuzzy logic system is composed of three components: fuzzification, fuzzy interference engine, and defuzzification as shown in figure(3.12). We exploit the power of Fuzzy logic in handling the uncertainty on making decisions to evaluate the capability of each robot and decide whether it is liable to participate in the recovery process or not.

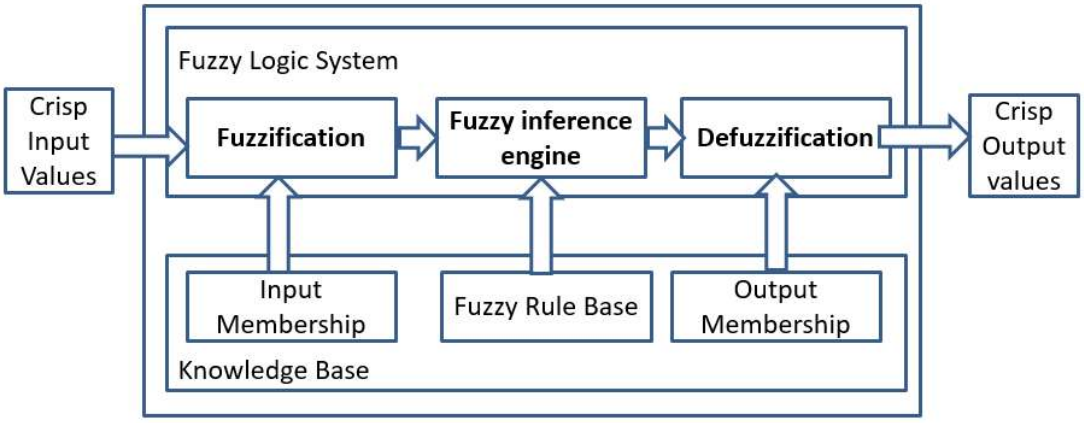


Figure 3.12: Fuzzy Logic System.

3.4.1 Fuzzification

It is the process of converting the system input and/or output data into fuzzy sets. We use the triangular curve type to shape our memberships, which is the most common curve types. Our three membership functions are depicted in Figure(3.13). The partition recovery supervisor is responsible for this process. Our

proposed approach focuses on the main factors that decide whether the node is liable to become a recovery team member or not. These factors are the distance from the CoD, the power level and the rank of the node. They are used by fuzzy logic as input memberships. The first membership is the distance between the node (i) and the CoD. The Node distance varies based on the deployment area. Hence, the distance membership is unbounded; this makes the membership more confused. To make the proposed algorithm more flexible, the measured distance values have been adjusted to a common scale, whatever the distance range is, the normalized distance range will have values over [-1,1]. The normalization is computed using the following equation.

$$\mathbf{x}' = \frac{x - min(x)}{max(x) - min(x)} \tag{3.3}$$

When NDist ∈[-1,0] the distance is defined as near (the shorter the distance, the higher priority), when NDist ∈[-0.5,0.5] the distance is defined as a medium, and when NDist ∈[0,1] the distance is defined as far. The second membership is the amount of energy that a node has; having a high level of energy helps the node to move longer without any danger of exhausting its power, which may cause the network to be partitioned again. When Power ∈[0,0.4] the level of power is defined as low which gives the node low priority in the movement, When Power ∈[0.1,0.8] the level of power is defined as a medium, and When Power ∈[0.6,1] the level of power is defined as high which means that the node has a great ability to move. The last input membership is the node rank; the lower the rank, the

lower is the expected number of nodes to move. Hence, this node is granted higher priority. This membership is very important, because if the node is moving while it is connected to many nodes, this will cause disruption to the network topology. When Rank $\in$[0,5] the rank is defined as Low, when Rank $\in$[2,7] the rank is defined as Medium, and when Rank $\in$[5,10] the rank is defined as High. Finally, the output membership, which is called priority, is expressed in six values to make it more accurate as VLow, Low, Medium, SlightHigh, High, VHigh. The top of the output priority list is chosen to be the recovery team leader.

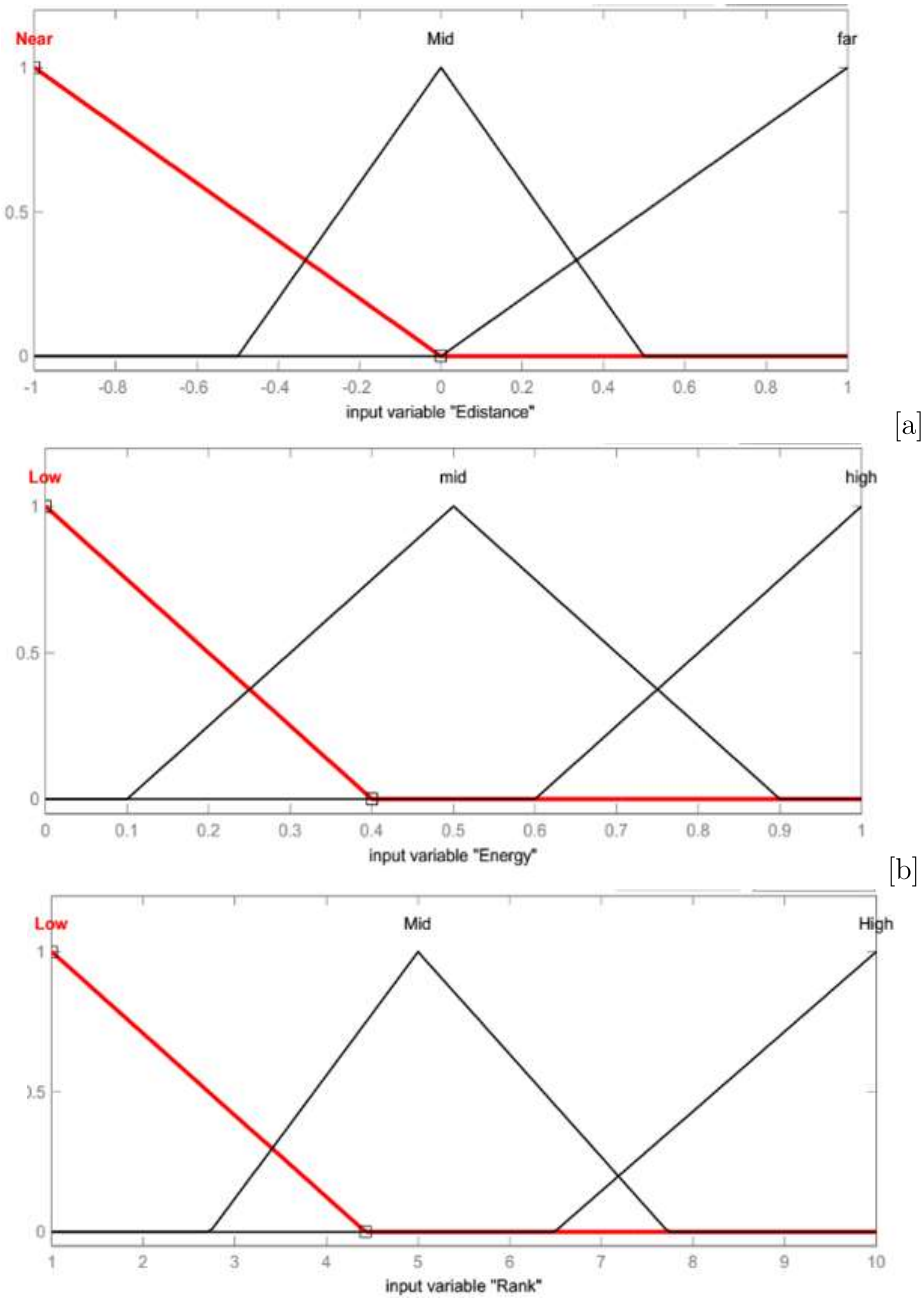


Figure 3.13: Input Fuzzy memberships.

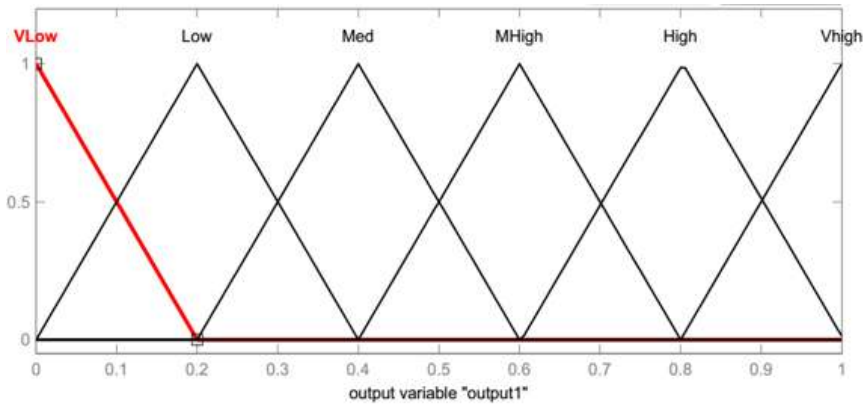


Figure 3.14: Output Fuzzy memberships.

3.4.2 Fuzzy Inference Engine (FIE)

FIE performs an approximate reasoning to find an output numerical value by associating the input memberships with fuzzy rules; this is also known as the rule firing in which the inputs are combined to produce the output. Inference rules are stated in form If x&y Then Z. As shown in the table (3.1), a list of 27 rules (three levels for each input membership) is produced for our problem; these rules will be used by the inference engine to generate its reasoning.

3.4.3 Defuzzification

The process of converting a fuzzy control action into non-fuzzy ones is called defuzzification. In our proposed approach, we use the centroid point method [36].

#	Distance	Energy	Rank	**Output**
1	Near	Low	Low	**Low**
2	Near	Low	Medium	**Low**
3	Near	Low	High	**Very Low**
4	Near	Medium	Low	**High**
5	Near	Medium	Medium	**Med high**
6	Near	Medium	High	**Medium**
7	Near	High	Low	**Very high**
8	Near	High	Medium	**high**
9	Near	High	High	**Med High**
10	Medium	Low	Low	**low**
11	Medium	Low	Medium	**V low**
12	Medium	Low	High	**Very low**
13	Medium	Medium	Low	**Med high**
14	Medium	Medium	Medium	**Med**
15	Medium	Medium	High	**Low**
16	Medium	High	Low	**High**
17	Medium	High	Medium	**Med High**
18	Medium	High	High	**Med**
19	Far	Low	Low	**Low**
20	Far	Low	Medium	**Very low**
21	Far	Low	High	**Very low**
22	Far	Medium	Low	**Med**
23	Far	Medium	Medium	**Low**
24	Far	Medium	High	**Very low**
25	Far	High	Low	**High**
26	Far	High	Medium	**Med high**
27	Far	High	High	**Med**

Table 3.1: Fuzzy interference rules.

3.5 Candidate list Preparing

The partition recovery supervisor is responsible for this process. The recovery team members are chosen carefully to minimize the travelled distance as well as the possible future failure. For example, cut-vertex robots are either not chosen as members in the recovery team or they should be replaced with another non cut-vertex robots. This list contains a set of robots that will move to restore the connectivity. In our proposed strategy, the candidate list members are prioritized using fuzzy logic to form the recovery team. Let R be the maximum transmission range between two nodes. To estimate the maximum expected number of nodes (M) needed to complete the recovery process, we divide the recovery distance by the maximum transmission range; this value is rounded up to the next largest integer as follows. The required recovery distance is not equal to the distance from the pivot point to the CoD (MD), but is equal to the subtraction of R/2 from it (RD) as in Equ.3.4. The Number of recovery nodes in Figure (3.15) that move to CoD is optimal because it is based on RD not in MD. If the CoD located at (s_x,s_y), then the distance from node (i) to the supervisor denoted d_i^s is can be computed as follows

$$\mathbf{d_i^s} = \sqrt{(x_i - s_x)^2 + (y_i - s_y)^2}$$

$$\mathbf{MD_i^s} = min(d_1, d_2, ..., d_n)$$

$$RD_j = MD_j - R/2 \tag{3.4}$$

$$\mathbf{M_j} = [\frac{RD_j}{R}] \tag{3.5}$$

For the jth partition, the supervisor node finds the pivot node that has the minimum distance to the CoD. Based on the node's state, they are divided into two main lists: cut-vertex list containing the nodes that cannot move, and backup list containing the nodes that will participate in the recovery process. The supervisor updates these two lists automatically. Whenever the node state changes due to the movement of its neighbors, the supervisor should be informed.

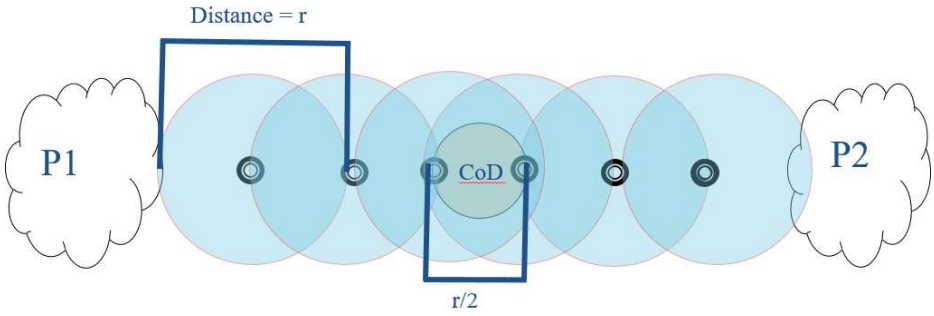


Figure 3.15: To get the connectivity back both P1 and P2 send their recovery team until they reach (CoD r/2), where (r) is the communication range.

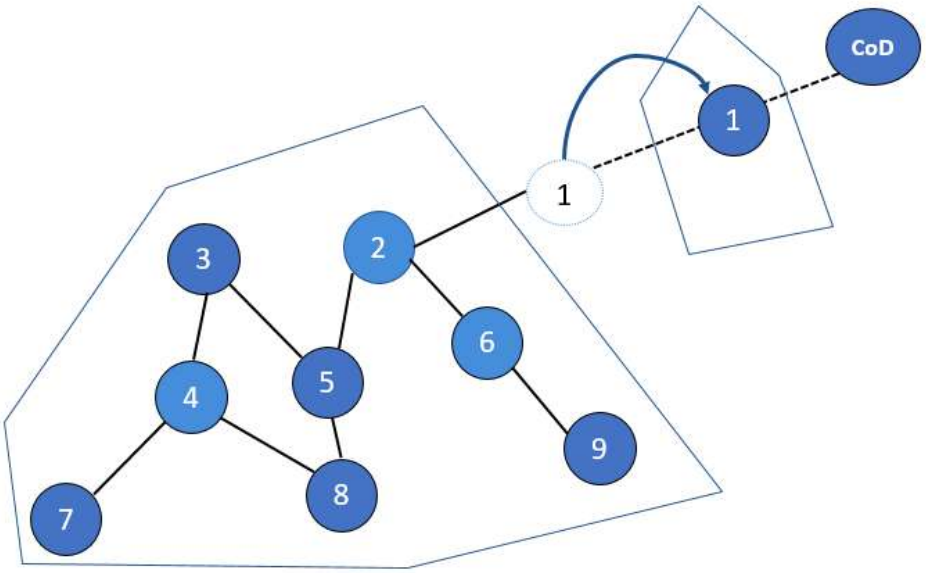
3.6 Candidates Selection approaches

Based on the priority list resulted using fuzzy logic, the leader will start moving toward the CoD while it keeps its communication with next node in the list (i.e.

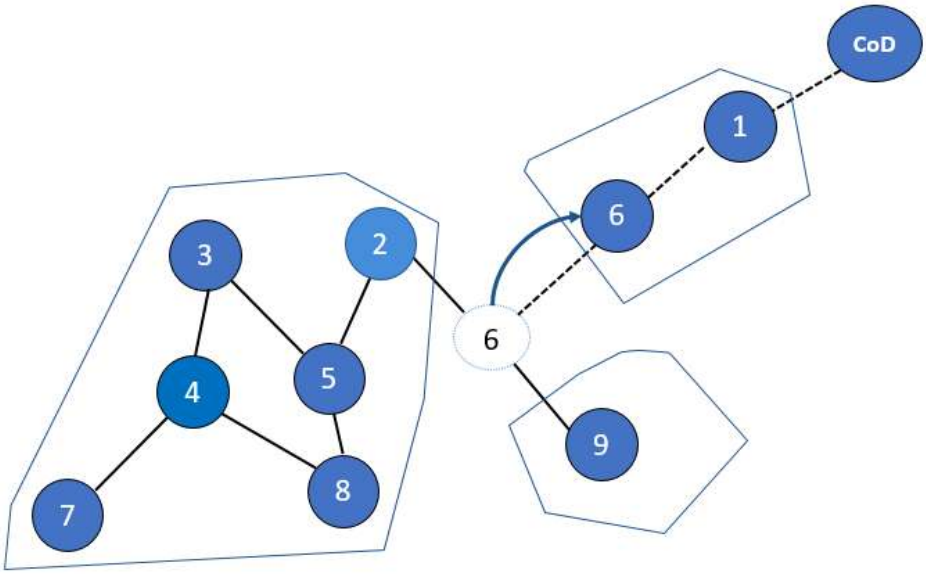
it will move for a maximum distance of R). If it gets a connection with the CoD or a node within another partition, it stops; otherwise, it will inform the next node to follow. The chosen of the next node could be achieved in many ways based on different criteria. In the following, besides our three proposed approaches the main two approaches are investigated.

3.6.1 Nearest Node

The chosen of the next node to follow is depend on how near is it from the CoD. This metric (Nearest node) is almost used in all the literature works as in [21] (DORMS). This approach has two fatal drawbacks. The first one is that the node is forced to move even if it dies. The second one is that the movement of the nearest node(a cut-vertex one) causes more partitioning in the network as it is illustrated in figure (3.16)



[a]



[b]

Figure 3.16: (Nearest node), an example how the movement of the nearest node causes more partitioning in the network.(a) when node 1, the nearest node,move one step forward the network get partitioned again. (b) node 6 movement splits the network again.

3.6.2 Most qualified node

The top nodes in the candidates list are considered the most qualified nodes. When some characteristics of the node are the more important than the others in the recovery process, the network will get partitioning over and over as an illustrated in Figure(3.17).

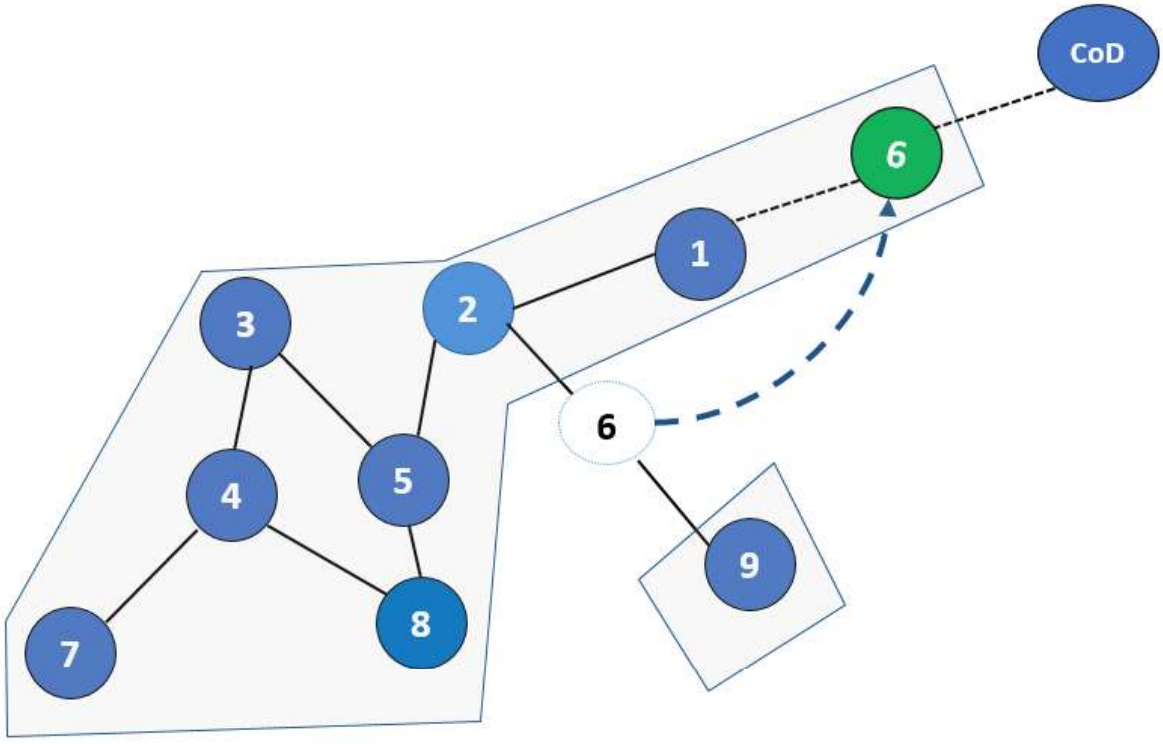


Figure 3.17: (Most qualified node), an example how the movement of the best node causes more partitioning in the network. Node 6 movement splits the network again

3.6.3 CoRFL

In CoRFL, we parameterize the selection of the recovery team according to fuzzy logic rules as in the most qualified node approach. Using CoRFL, cut-vertex nodes do not participate in the recovery process to minimize the intra-segment partitioning. As it is shown in Figure(3.18), leader and its team will keep going one after another until they reach the CoD, while cut-vertex nodes (in red) do nothing. Backup, cut-vertex, and moving lists are updated after every move. In order to keep the network connected, the cut-vertex node does not participate in the recovery process even if it is the most qualified node.

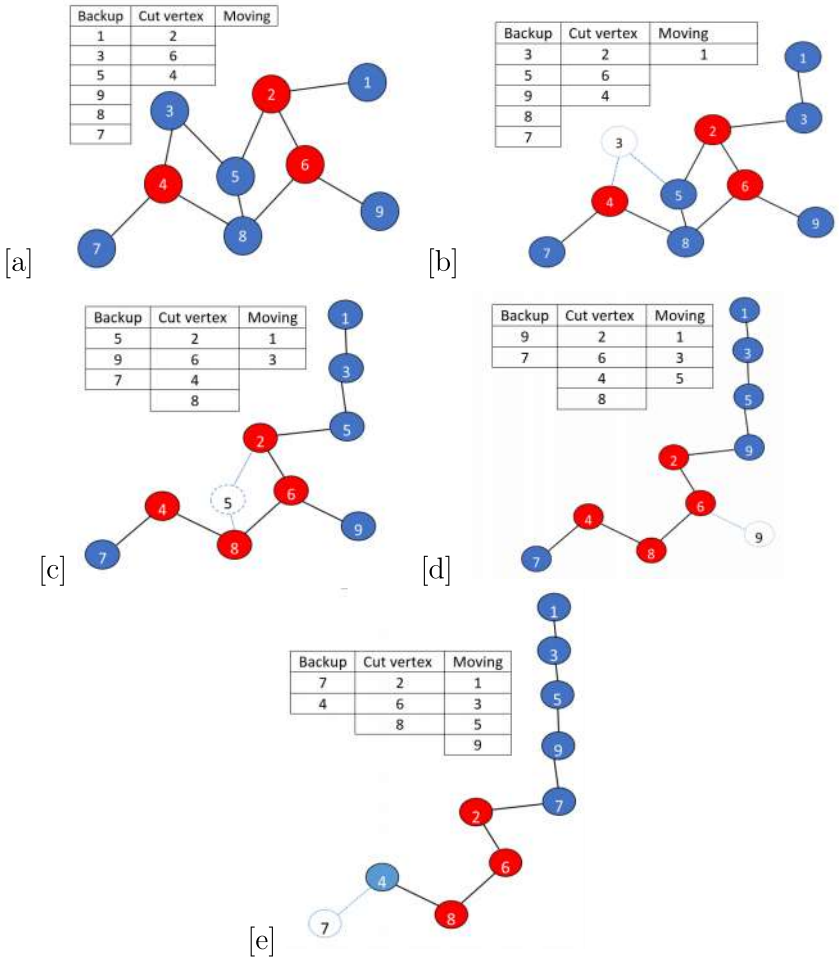


Figure 3.18: A detailed example, (a) the initial partition, (b-e) cascaded movement.

3.6.4 CoRFL2

CoRFL2 is similar to CoRFL with cut-vertex replacement using fuzzy logic which is presented in figure(3.19) algorithm.

```
function CoRFL2_replacement(Nodes, replaced_node)
 Identify the replaced node
 Prioritized_Nodes = fuzzy_logic(Nodes, replaced_node)
 Prioritized_Nodes= sort(Prioritized_Nodes)
 m = Count(Prioritized_Nodes)
n = log2(m)
solution_found =False
do
     Generate all the possible solutions (n)
     Exclude the solutions that include nodes with limited power.
    If solution has nodes with limited power.
        delete(solution)
    if solution is invalid (makes more partitions)
        delete(solution)
   if length(solutions)> 1
      solution_found=True
  else
     n=n+1
while solution_found=True

solutions_list =fuzzy_logic(solutions)
best_solution= solutions(1) // the first one
end function
```

Figure 3.19: CoRFL2 replacement algorithm.

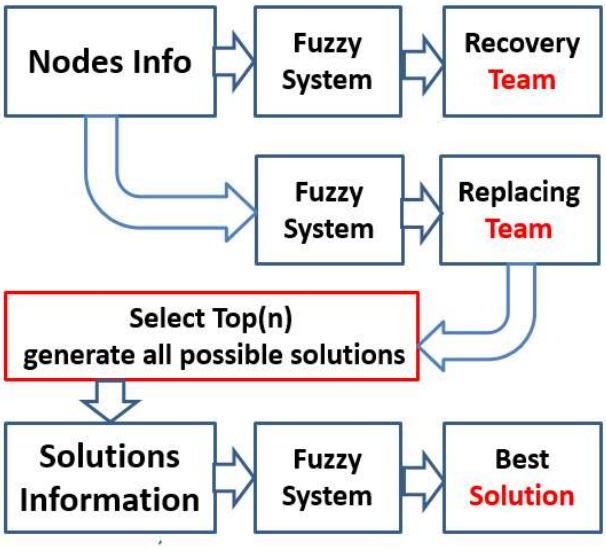


Figure 3.20: CoRFL2 Fuzzy systems.

When the most qualified node is a cut-vertex, and its movement will split the network into more partitions as illustrated in Figure(3.17), the node(s) with high power and least travel distance(s) will be selected to replace it as illustrated in figure(3.22). As shown figure(3.20), CoRFL2 uses fuzzy logic inferencing three times, in choosing the recovery team list, in choosing the replacement team for a cut-vertex node if necessary, and in deciding which replacement solution is the best. The same fuzzy logic system used in choosing the recovery team is used in choosing the node replacement team . Partition supervisor have the information of the partition nodes , then it uses fuzzy logic to prioritize the replacement team. As shown in figure(3.21), nodes distance to the replaced node (6) , their rank, and their power is the input of the fuzzy logic.

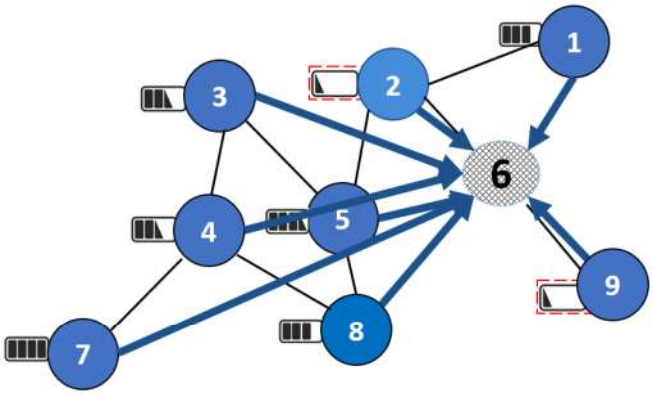


Figure 3.21: CoRFL2 replacement team information gathering, building the inputs of the fuzzy logic (nodes distance to the replaced node and their power and rank).

After prioritizing the replacement team, cut-vertex replacing node(s) could be one, two, three or more. So the process of finding the best number replacing nodes goes through many steps which we will explore in the following.

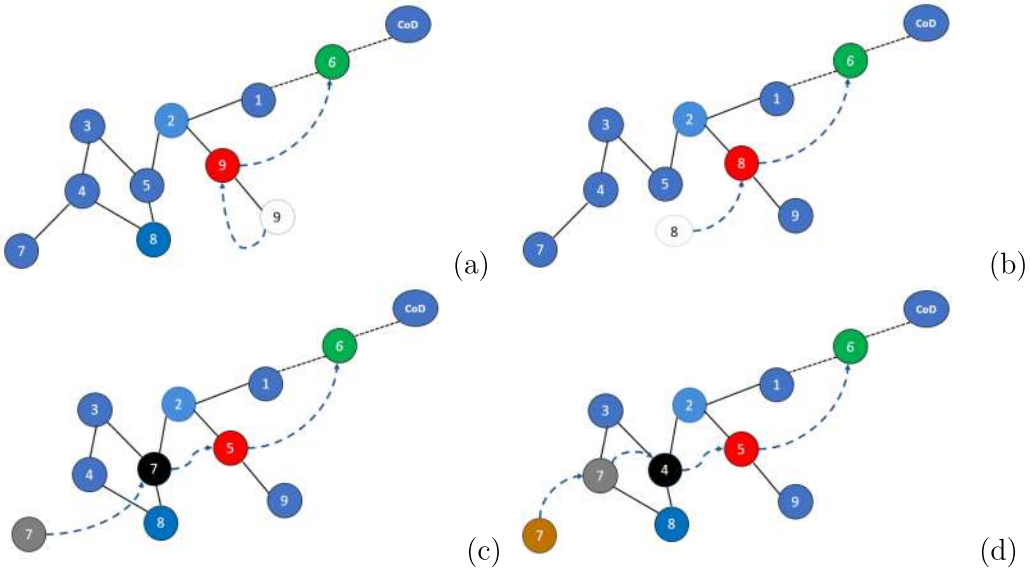


Figure 3.22: Four different replacements possibilities to solve the problem shown in figure(3.17). (a and b) the replacement is done smoothly unlike the others where node (5) place movement cause another problem. (c-d) shows how the replacement process done in an iterative way.

1. Based on the number of the nodes there are 2^n different possibilities of choosing these nodes to achieve the recovery process to replace the cut-vertex nodes if necessary. If the number of nodes (m) is large , then from the priority list of nodes the top (n) would be enough . The value of ,n , is getting in an incremental way.

$$\mathbf{n_0} = log_2(m) \tag{3.6}$$

Where the initial value of n (n_0) is getting by Equation(4.4) and it keeps growing until the chosen nodes have at least one uncut-vertex node with

enough power or two with moderate power. When there are enough nodes to do the job which are the top (n) nodes in the candidate list, the replacement possibilities is generated . As shown in figure(3.23), to illustrate the idea where n=5 so there are $2^5 = 32$ possible solution. The columns represents the nodes and the rows values express state of participating of the node. If the value is (1) it means the node will move , and if it is (0) it means the node do nothing.

0	**0**	**0**	**0**	**0**
0	0	0	0	1
0	0	0	1	0
0	0	0	1	1
0	0	1	0	0
0	0	1	0	1
0	0	1	1	0
:	:	:	:	:
1	1	1	0	0
1	1	1	0	1
1	1	1	1	0
1	1	1	1	1

Figure 3.23: The most qualified node (6 in green) is a cut-vertex and its movement will split the network into two parts.

2. Not all the nodes are ready and able to participate in the replacement process especially those who are running out of power. When the node suffers from shortage power, it may not be able to reach the target position which affect the whole recovery process and the future network operation. So nodes with

limited power should not participate in the replacement process, otherwise, they are considered as replacing nodes as it is illustrated in Figure(3.24). As a result for that, not all the possible solutions in figure(3.23) are valid especially those who force node two to move while its dying as an example. Figure (3.25) shows the remaining possibilities when node (two) do nothing.

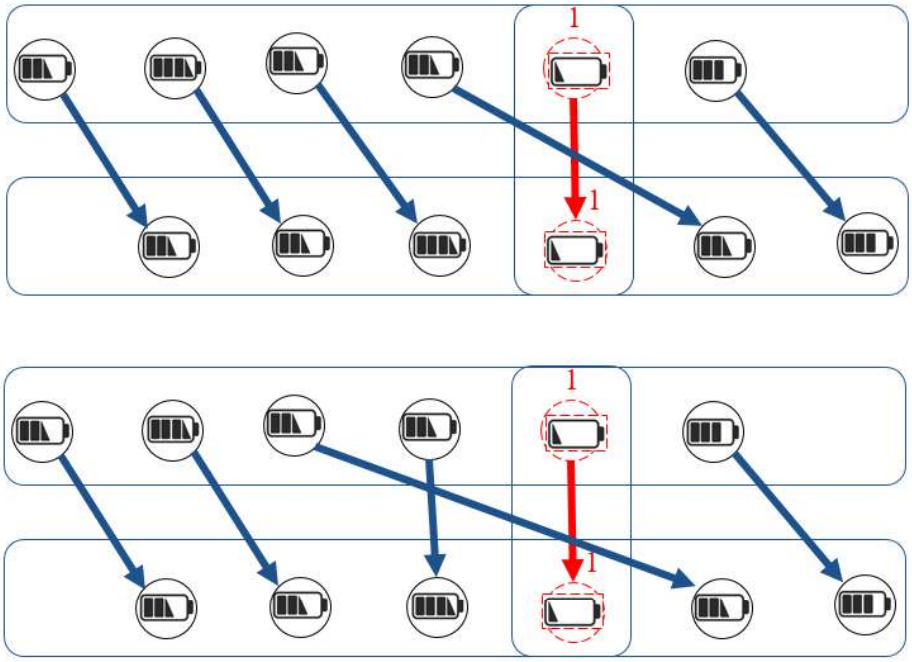


Figure 3.24: Two possible replacing solutions while node (2),with limited power, stay where it is and the others participate in the replacement process.

3. The remaining possibilities are refined by checking each path consequences. If the path splits the network again or make the network at risk of new partitioning it is not a good path, so it is ignored. The valid replacement possibilities paths called Refined-solutions.

0	0	0	**0**	0
0	0	0	**0**	1
0	0	1	**0**	0
0	0	1	**0**	1
0	1	0	**0**	0
0	1	0	**0**	1
0	1	1	**0**	0
0	1	1	**0**	1
1	0	0	**0**	0
1	0	0	**0**	1
1	0	1	**0**	0
1	0	1	**0**	1
1	0	0	**0**	0
1	1	0	**0**	1
1	1	1	**0**	0
1	1	1	**0**	1

Figure 3.25: All the possible replacing solutions while node (2) not participate.

4. Total-travel distance and the average level of power of the moved nodes, for each solution, is computed .

5. Fuzzy logic system is used to determine which solution is the best. Both the total-traveled distance and remaining power are input memberships. And the output membership, which is called priority, is expressed in four values as Low, Medium, High, VHigh. The solution which has the best value in the output priority is chosen to fulfill the replacement process. Fuzzy memberships are shown in figure(3.27), and the fuzzy system rules are shown in table(5).

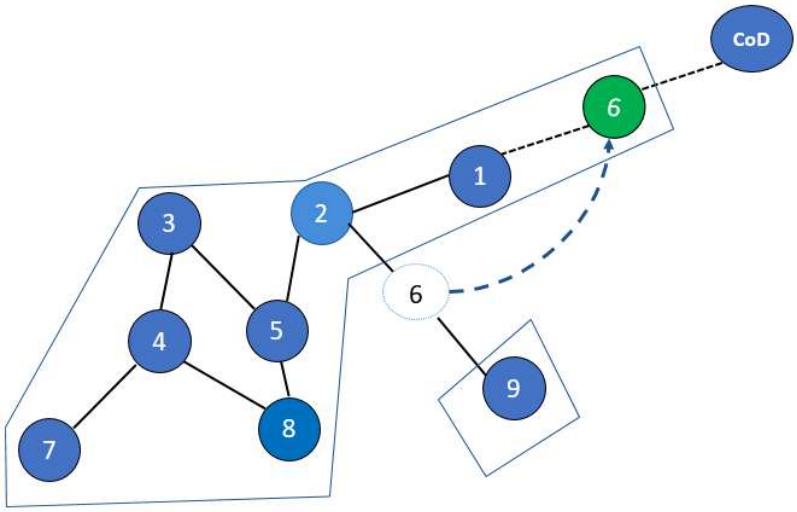


Figure 3.26: The most qualified node (6 in green) is a cut-vertex and its movement will split the network into two parts.

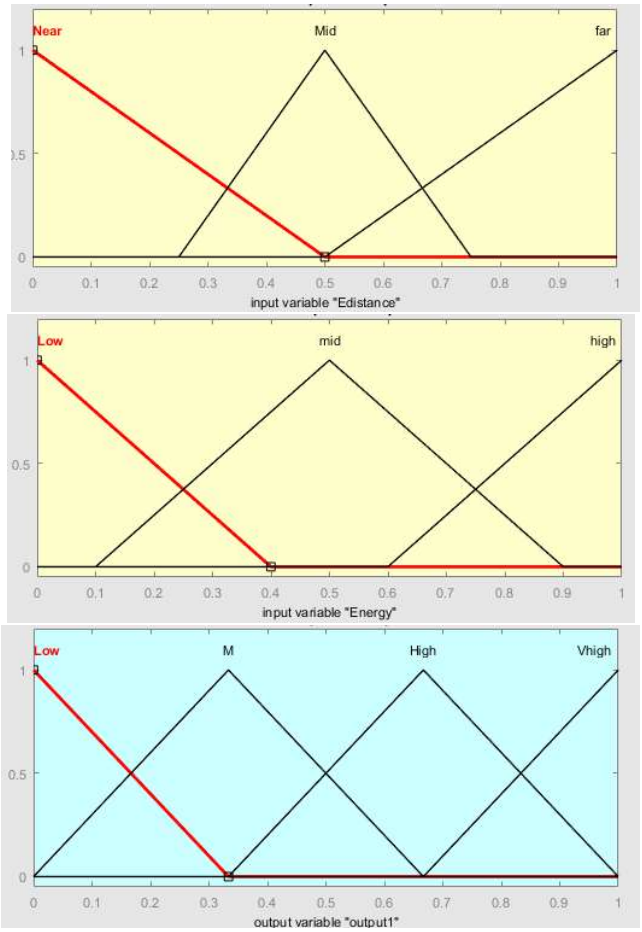


Figure 3.27: Fuzzy memberships of the replacement process, two input and one output.

#	Total-traveled distance	average power level	output (Priority)
1	Near	Low	Low
2	Near	Medium	High
3	Near	High	Very high
4	Medium	Low	low
5	Medium	Medium	Medium
6	Medium	High	High
7	Far	Low	Low
8	Far	Medium	Low
9	Far	High	Medium

Table 3.2: Fuzzy interference rules.

The best cut-vertex node is replaced by following the obtained solution from the fuzzy system.

3.6.5 CoRFLN

CoRFLN is similar to CoRFL with simple cut-vertex replacement. In CoRFLN the cut-vertex node is replaced by its nearest node. Cascade replacement is needed when the replacer and the replaced nodes are cut-vertex as shown in figure(3.28).

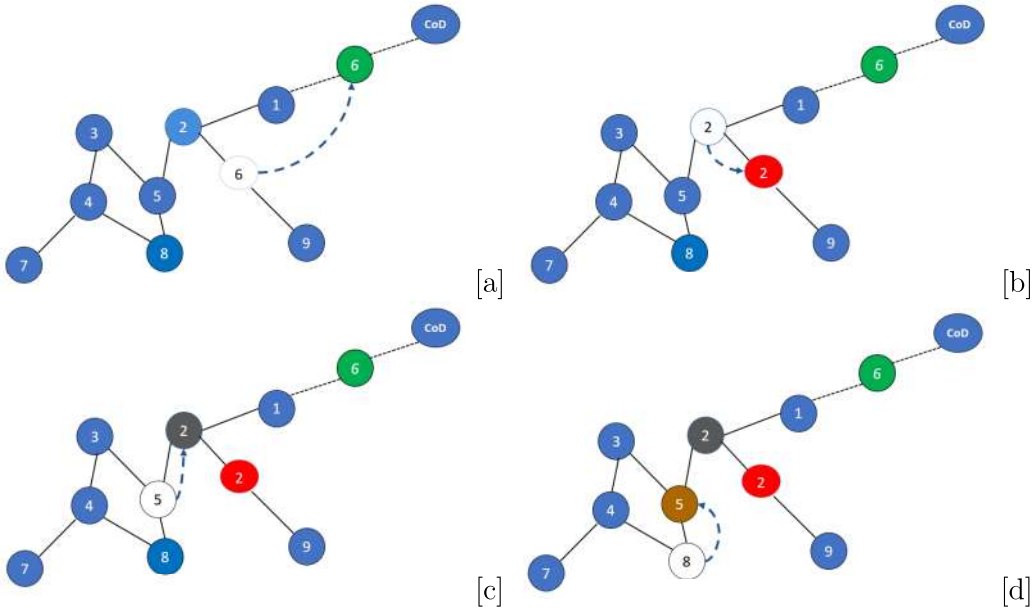


Figure 3.28: CoRFLN replacement illustrative example.

3.7 Partitions leaders encountering

During the cascaded movement of the recovery team, they might be got connected with another partition. To complete the recovery process, this connection has three possibilities:

- When the recovery teams from different partitions are connected with each other before reaching the CoD, the leaders will alternate the movement till they reach the CoD as shown in Figure(3.30) (i.e. the recovery team members will move toward each other to got 2-connectivity).

- When the recovery team is connected with another partition and that partition is closer to the CoD, the closer partition will be in charge

- The recovery team does nothing if they get connected to another partition which is already has a hand to the CoD.

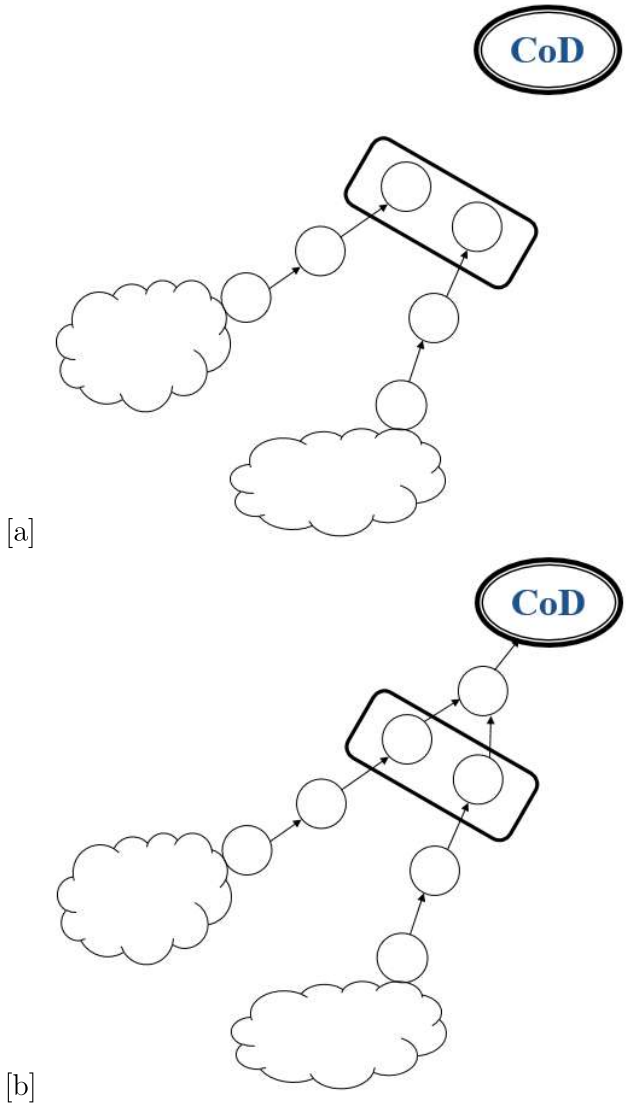


Figure 3.30: (a) Alternating the leaders, (b) leaders alternate sending nodes after the connection is done ,they move toward each other to establish two connectivity between the recovery teams.

3.8 Simulation setup

The connectivity restoration simulator has been developed using MATLAB. A connected WSRN is deployed randomly in an area of size 200m x 200m. The CoD is located at (100,100) in the simulated area. Every point in the performance curves is produced using the average of 20 different random topologies.

3.8.1 Assumptions

The following enumerates the key system model assumptions:

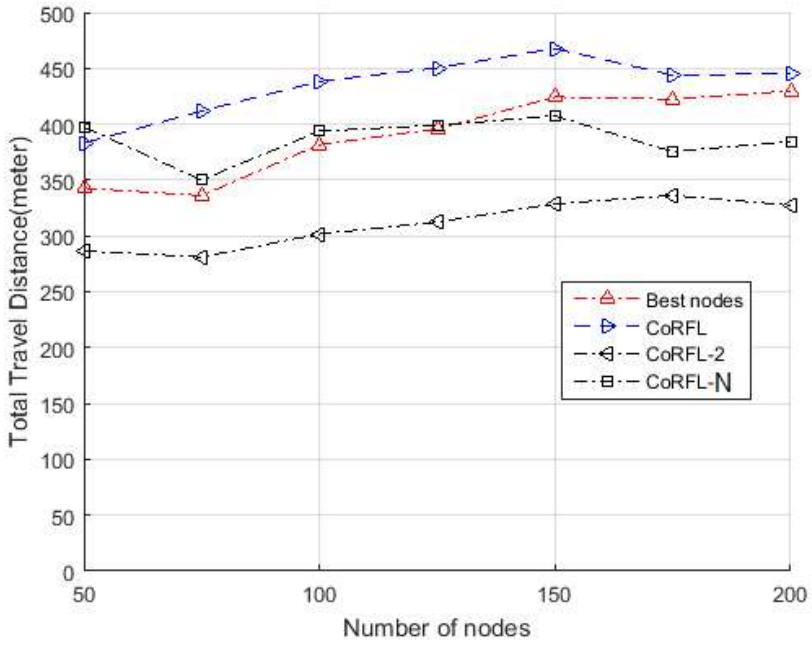
1. Robots are homogeneous, i.e., have the same speed, service capabilities, energy supply, etc.
2. Each Robot knows its initial location.
3. Network communication range is fixed.
4. The CoD is known to the robots.
5. The robots have the capability to sense how danger their surroundings.
6. Localization information is perfect.
7. All exchanged information is transmitted over a secure channel.

3.8.2 Performance metrics

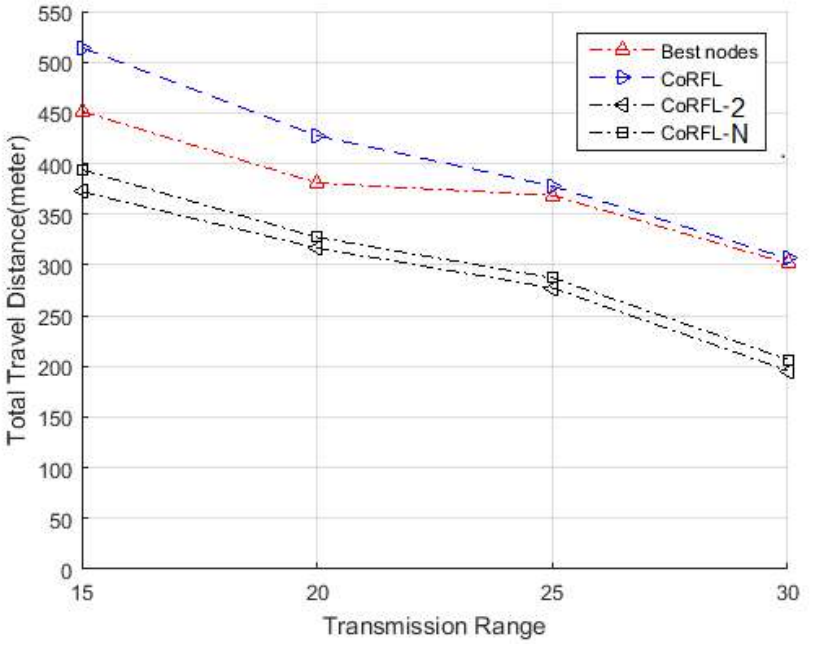
To assess the performance of the proposed approaches , the following WSANs metrics have been used. The following metrics setting is either with a fixed transmis-

sion range (R=15), or varying transmission range with fixed network size (n=200).

- **Total Traveled Distance:** This metric reflects the total movements of recovery nodes, which is considered as the total cost of the recovery process as the more movements, the more power consumed. Most of existing approaches in the literature did not explain how to deal with cut vertex nodes that will divide the network again except CoRFL. The fact is that the total travelled distance is not based only on the total movements of the recovery team but also it includes all necessary movements to keep the inner partition connectivity functional. The total traveled distance of our approach is depicted in both Fig.13 and Fig.14. We compared CoRFL2 to its previous version CoRFL, to the most common way in the literature that is based on the nearest node should move, and to the way that base on the most qualified node should move. We did not show the results for a specific algorithm in the literature as they did not explain how inner nodes will move to keep the connectivity functional.

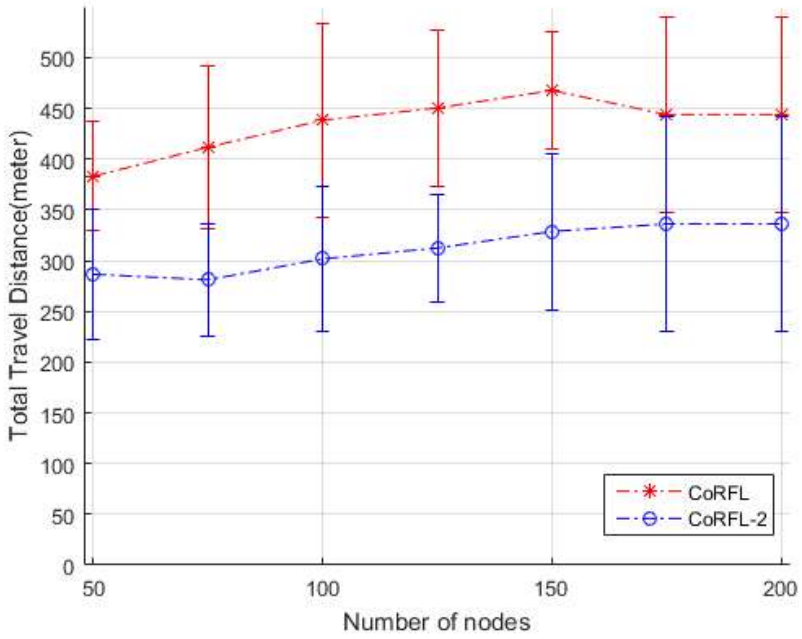


[a]

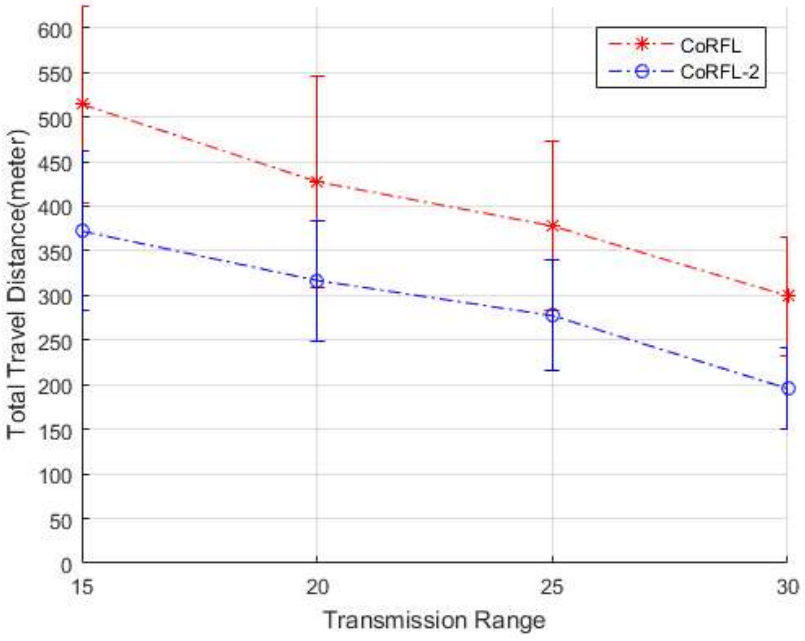


[b]

Figure 3.31: Total Travelled Distance.



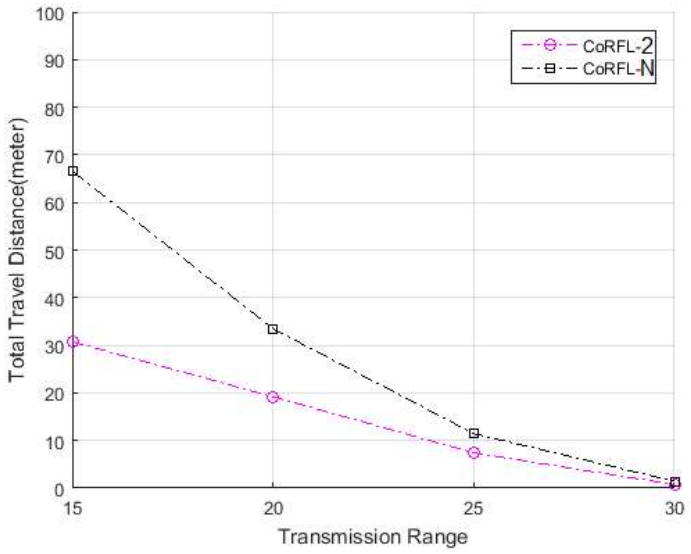
[a]



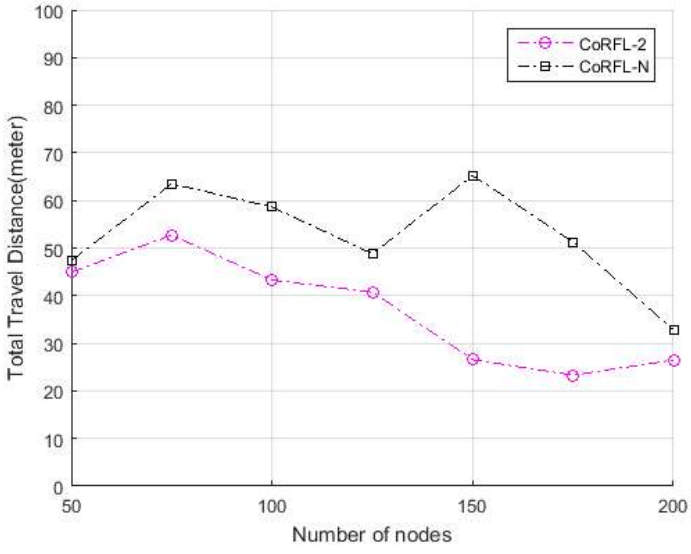
[b]

Figure 3.32: Total Travelled Distance confidence interval for both CoRFL and CoRFL2.

- **Replacement total travel distance:** This metric concerns about the total travel distance required to replace a cut-vertex node, the simulation result is shown in figure(3.33) .



[a]

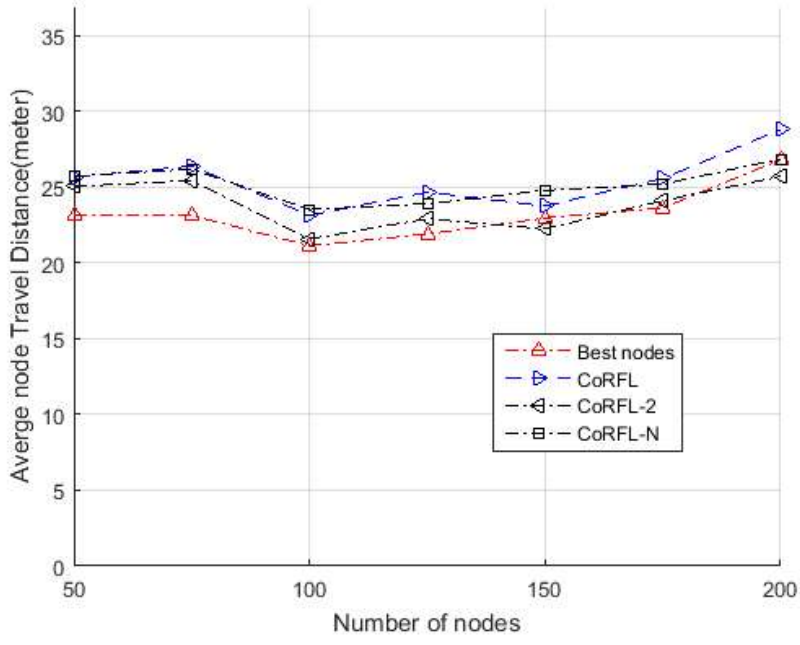


[b]

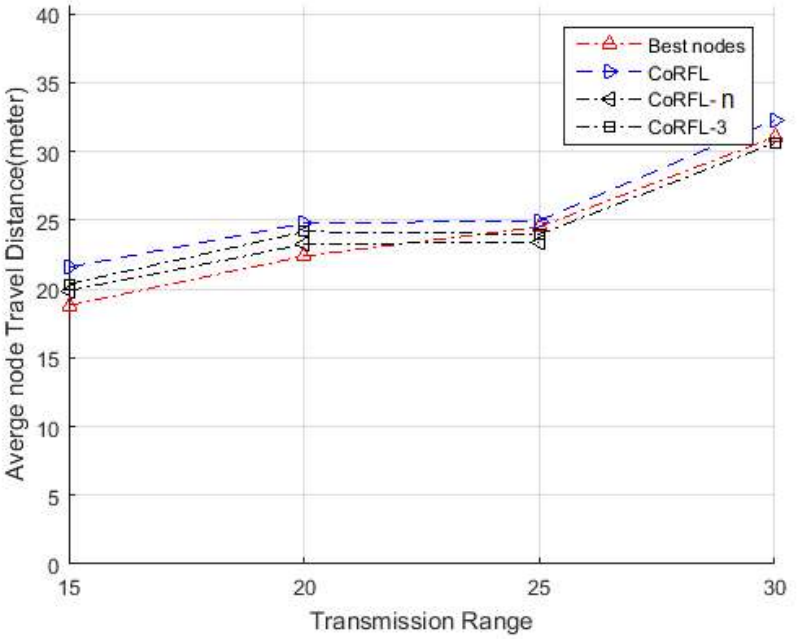
Figure 3.33: Replacement total distance.

- **Average Movement per Node:** This metric concerns about the average

movement per node, the simulation result is shown in figure(3.34).



[a]



[b]

Figure 3.34: Average movement per node.

- **Traveled distance per partition:** The network connectivity restoration is getting worse when the number of partitions is high. Figure(3.35) shows how good CoRFL2 is. CoRFL2 prove that it is the best most of the times. Even though CoRFL family algorithms do not focus only on the distance , but they still provide a good result comparing to DORMS.

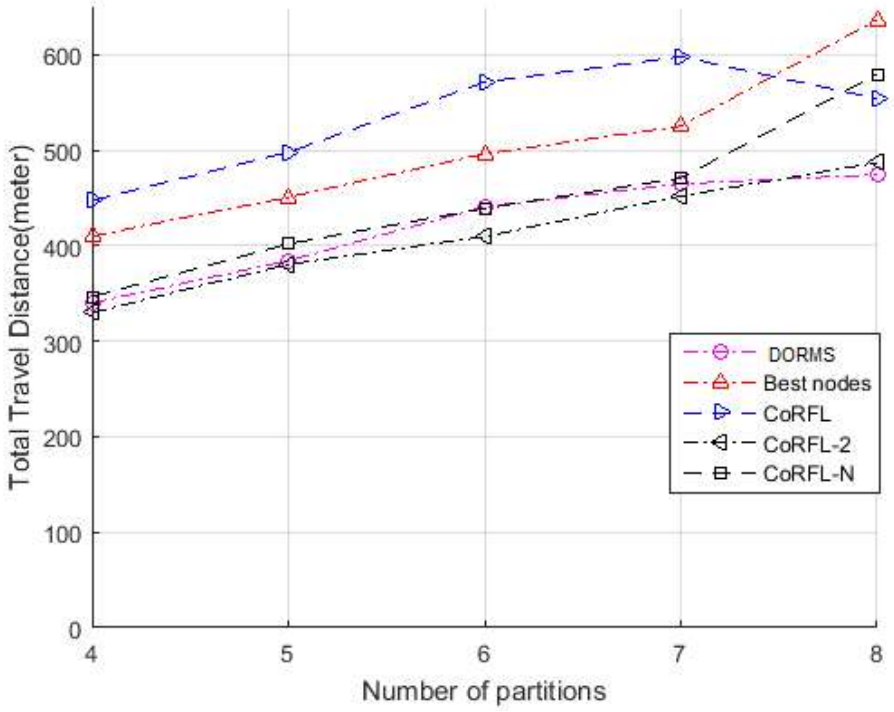
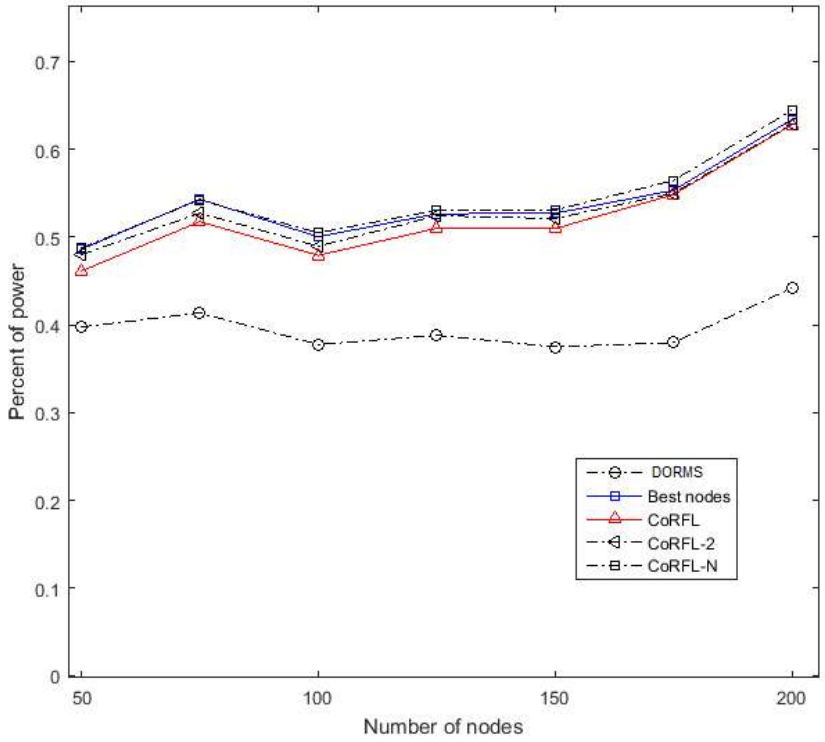


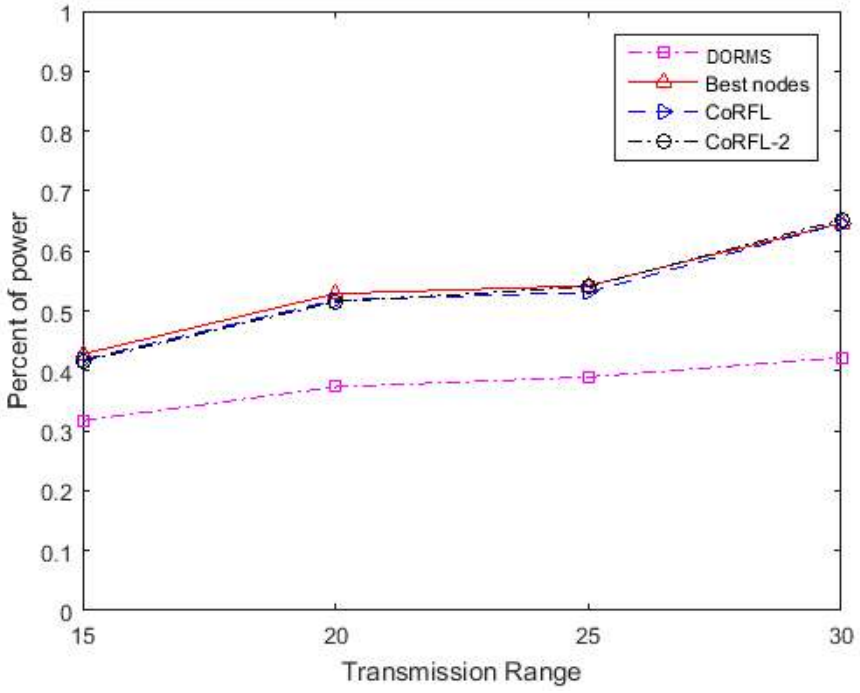
Figure 3.35: Total travelled distance per number of partitions.

- **Recovery Team Power:** This metric is very important to test the ability of the recovery team to sustain the established connectivity and mitigate the future failure due to losing network power. The higher the power level, the longer is the expected network lifetime and the robust is the deployed network. The previous recovery approaches do not pay much attention to

this issue. We compare the performance of CoRFL2 with CoRFL, Nearest node(DORMS), and best nodes in terms of travelled distance, and the average power of the recovery team. We can observe from figure(3.36) that the power of the recovery team established by CoRFL2 has higher power than the one established by the others. This result ensures that CoRFL2 team can live and sustain network connectivity for a longer period.



[a]



[b]

Figure 3.36: Average power capacity of the recovery team.

3.9 Conclusion

In this work, we have investigated the WSRNs connectivity restoration after a simultaneous nodes failure. The proposed solution is based on repositioning the most qualified nodes in each partition toward the CoD in a distributed manner. The recovery nodes are selected using fuzzy logic based on their level of power, node rank and the distance. The simulation results have confirmed the effectiveness of our proposed approach compared to DORMS. Our proposed approach not only federates the partitioned network, but also minimizes the future failure and minimizes the possibility of future disruption within each partition.

CHAPTER 4

PATH PLANNING

In fact, one of the main steps in the process of connectivity restoration is to find the best path for the robots to follow in order to accomplish the recovery. The number of robots required to fulfill the connectivity restoration is determined by the length of the path influenced by the changes in terrain, environment, and obstacles. Path planning has been extensively used in different fields, namely, robotics, games, manufacturing, automotive applications, and aerospace applications, etc. Generally, the critical step in path planning is to find the best collision free path from the start position to a defined goal position through a known, unknown or partially known environment and to optimize it with respect to some criteria. Obviously , the environmental factors significantly determines the path planning algorithm and determine the scope of the problem.

4.1 Motivations for Path Planning

The possibility of deploying the networked robots in a flat environment with no terrain differences and hazard-free is unusual in practice. What is more, certainly the partitioning problem occurs due to unexpected events in the robots deployment area. If there is no environment damage or obstacles the recovery team can use a simple straight path to reach the CoD. Most published recovery techniques the path planning as they assume the straight path is hazard-free.

Meanwhile in terms of environment reality, a restoring connectivity approach that takes into account the terrain is proposed in [5] . In [5], direct path is not assumed, but a static environment -where the environments is completely known and no new changes will occur- is assumed which is not always possible. A* [11] algorithm, which is widely used to find the least-cost path from a given initial location to the target location, is used in [5] to plan the recovery path in the assumed completely known environment. This work has two main drawbacks:

- The Assumption of a static environment after a massive damage to the network is not practical in the real world.
- The assumed complete information about the environment and the terrain types and the new emerged obstacles is unjustified as it does not necessarily reflect the nature and the accumulated changes in the deployed area.

What is actually happening is that the network failure happens because of a hazardous event that devastates the area. These events introduces topological

constraints for the robot motion. Thus, the scope of the problem under investigation can be classified as a hybrid path planning due to the presence of information about the environment before the failure and the lack of information about the environment after the failure. However, before we proceed to our hybrid approach, variants of D* and Artificial Potential Field will be discussed first.

4.1.1 D* and its variants

One of the most studied and improved algorithm is D* (dynamic A*) proposed in[40] by Stentz, which overcome the shortcoming of A* algorithm in dealing with unknown or partially known environments to re-plan if necessary. D* incrementally repairs the path as any change in the environment is detected.

Many D* variants proposed in the literature attempting to fulfill the dynamic path planning. In implementing Focused D* algorithm -one of D* family algorithms which is proposed in [46]- on Khepera IV robot, we conclude that this family of algorithms work theoretically fine , but they have some practical limitations such as:

- Although their strict cell size solves the problem of finding the shortest path, it overlooks the actual size of the robot and obstacle. As illustrated in figure(4.1.a), D* finds a path that passes through the corners of the two red obstacles. This path is not valid and once the robot reaches close to the intersection point it recognizes this intersection as a new obstacle located at the black cell and consequently a new path is generated. This does not

only waste time but also waste energy. However, taking the actual size of the robot saves both time and energy.

When two neighboring points (in red) in the generated path share the same surrounding obstacles, then it is more practical to consider them as obstacles (in blue) as well. See Figure(4.1,b-c).

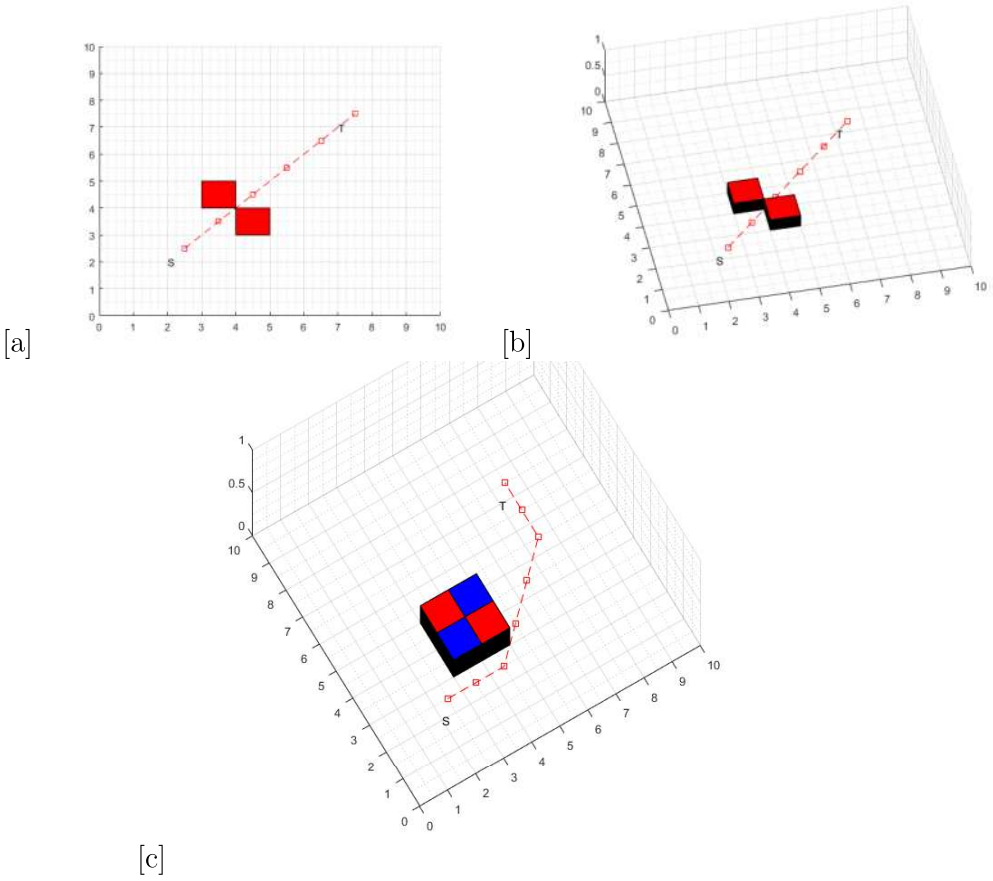


Figure 4.1: (a) D* finds a path that passes through the corners of two obstacles which is not a valid path as it shown in (b) it is a closed path, and (c) shows how this problem is solved.

- Rigid paths force robots to move diagonally along the shortest path regard-

less of their actual size and therefore collisions with the corner of obstacle inevitably occur. As can be seen in figure (4.2), collisions occur because the robot sensing system indicate the absence of obstacles in its next step and consequently this lead to two critical scenarios:

- the robot might **stop** at the corner while its wheels keep rolling without any progress.
- the robot might experience unplanned change in direction because of the collision and hence a localization problem occur.More importantly, the robot will most probably not recognize this drift.

As shown in figure(4.2), the path width (P_w) has to be greater than the width of the robot (R_w) plus a small tolerance (ϵ) which is necessary to prevent the contact between the robot and the obstacle.

$$P_w = R_w + \epsilon$$

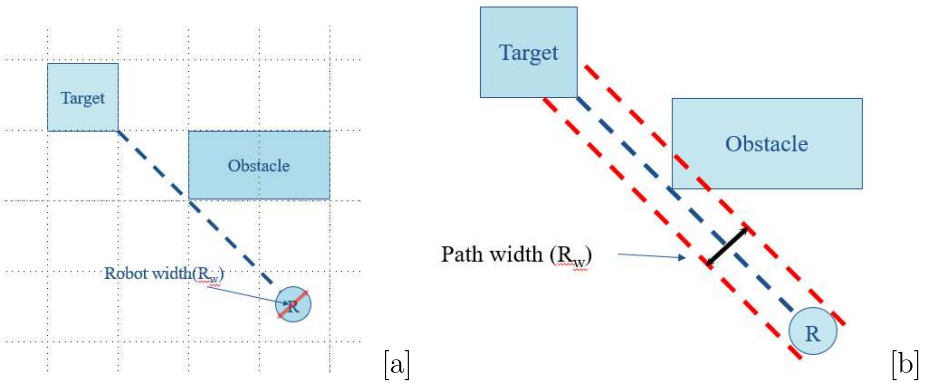


Figure 4.2: Robot size problem.

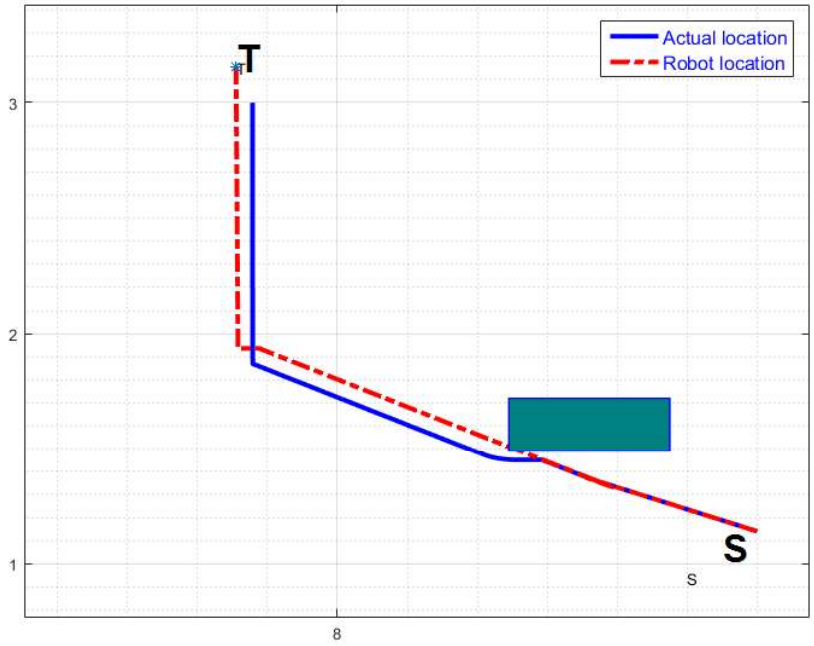


Figure 4.3: Robot size issue real experiment.

Figure (4.3) shows how the robot predetermined path (in red) is affected when it encounters an obstacle. The collision between the obstacle and robot leads to unplanned change of direction and hence a localization problem occurs.The key issue is that the robot does not recognize this drift and consequently the difference between the calculated trajectory(in red) and the actually travelled distance (in blue) will lead the robot to stop not at the target point rather at a different point. As a matter of fact, the robot count the number of turns of its wheels and considers it as the actual distance while not every turn is transferred into an actual movement. Moreover, if each turn is a real movement ,the robot might not follow the predetermined

trajectory.

- Obstacle size estimation is one of the main issues in dynamic path planning. The more information about the actual size of the obstacle, the better is the obtained path . This problem is a result of other two problems, namely, the limited accuracy of the robot sensors , and the representation method of the area under study.

- Obstacle localization : Robots do not have inherent abilities to instantaneously analyze our surroundings and know where they are in the context of their surroundings. In order to localize itself (to know where it is), a robot needs a good map and a perfect localization technologies and methods.

The last two issues cannot be resolved completely as robot sensors, motor accuracy, and localization method play the main role however minimizing their effects can be achieved. Artificial Potential Field is one of the best algorithms to tackle such problems and therefore it is in order to explore the Artificial Potential Field in the following section.

4.1.2 Artificial Potential Field

A well-known online path planning technique called Artificial Potential Field (APF) is one of the perfect methods to avoid obstacles . The essence of potential field is taken from nature. The idea is that depending on the strength of the field, or the slope of the hill, the particle, or the ball can arrive at the source of the field, the magnet, or the valley in this example.In robotics, the same effect is

simulated by creating an APF that attracts the robot toward the goal and repels it away from obstacles [14] . If the robot approaches the obstacle, a repulsive force is applied on the robot pushing it away from the obstacle. Figure (4.4) is to illustrate the idea behind the attractive and the repulsive force.

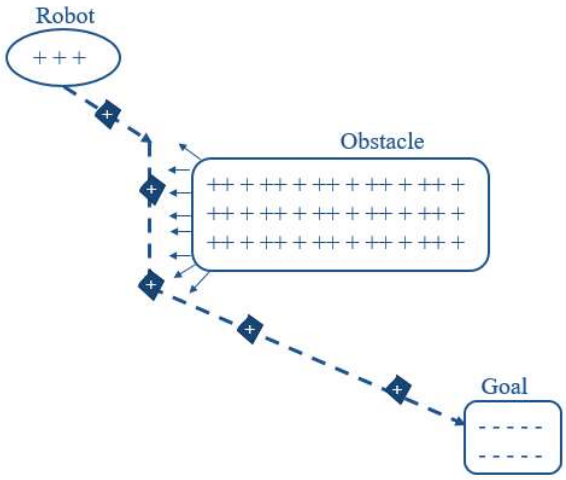


Figure 4.4: Positive charges repulse the robot and negative charges attracts it resulting the path denoted by a dashed line.

APF function is defined as the sum of an attractive force $U_{attr}(q)$ pulling the robot towards the goal and a repulsive potential $U_{rep}(q)$ pushing the robot away from the obstacles. According to [8] , the overall potential field is obtained by the following equations.

$$\mathbf{U_{attr}}(q) = \alpha \|q_i - q_g\|^2 \tag{4.1}$$

$$\mathbf{U_{rep}}(q) = \beta \ \|q_i - q_o\|^{-2} \tag{4.2}$$

$$\mathbf{U}(\mathbf{q}) = U_{attr}(q) + \sum U_{rep}(q) \tag{4.3}$$

$$\mathbf{F} = -U(q) = (U/x, U/y) \tag{4.4}$$

Where q_i, q_g and q_o are robot , goal and obstacle positions, respectively. The values of α, β are usually chosen to be less or equal to one. Where $\| \ \|$ is the Euclidean distance.

Artificial potential field path planning is very bad most of the times as its generated path in the presence of an obstacle is very long. Figure (4.5) shows that, APF path is very long , unlike D* path.

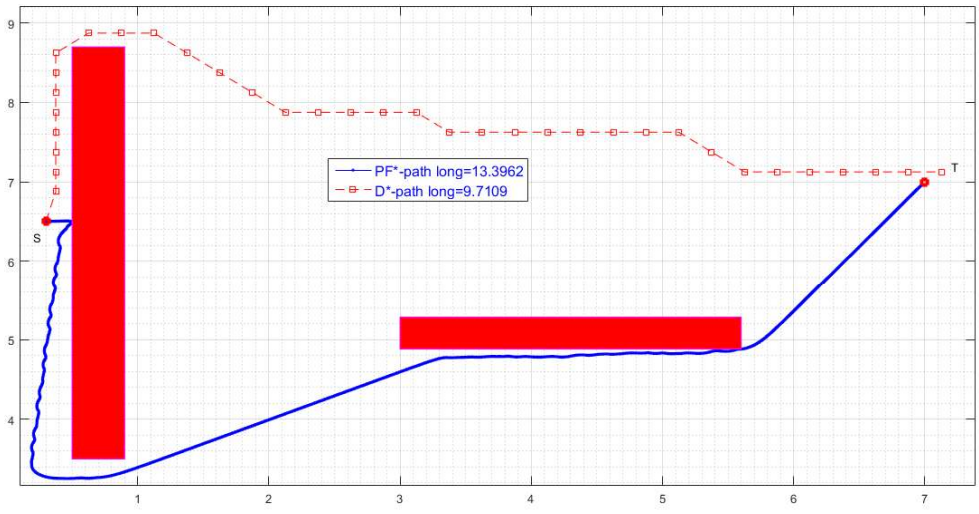


Figure 4.5: Potential Field path planning compered to D* .

4.2 Hybrid APF and D* (PD*)

Both D* and APF have their own shortcomings. Combining them will make the path much better and help the robot to avoid the corners of encountered obstacles. Therefore , we exploit the power of APF in avoiding the obstacles to smoothen the path generated by Focused D*.

Our experiments have been conducted using Khepera IV robot. The area is divided into cells, and the smaller the cell size the shorter the path we get. The following are the problems that we encounter in identifying the grid cell size.

- Khepera IV robot distance sensors are not accurate. Their values fluctuate too much which makes identifying the obstacle location hard. A real experiment as been conducted using Khepera robot to test the measurement accuracy. While the robot and the obstacle are resident, the robot distance sensors give a fluctuated measurements as illustrated in figure(4.6).

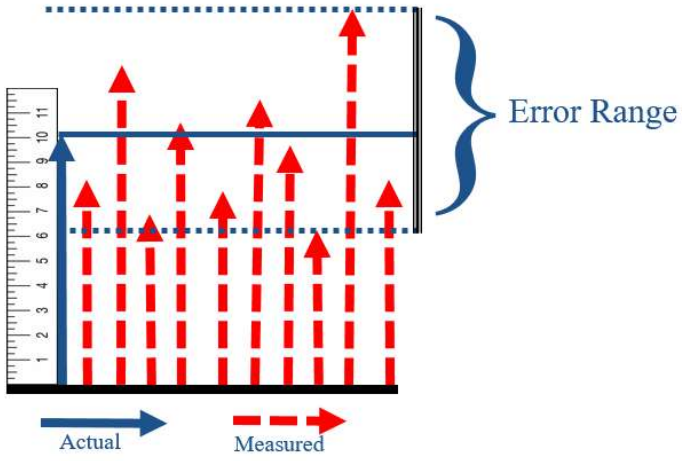


Figure 4.6: Real experiment of distance measurements using Khepera IV robot.

We found that, distance sensor error range ϵ is approximately equal to $\pm 4cm$.

- If the the cell size is smaller than the robot, the obtained path is untraversable if it goes through two obstacles as it is the shortest path as shown in figure (4.7). The robot could not move along that path because it is not wide enough.

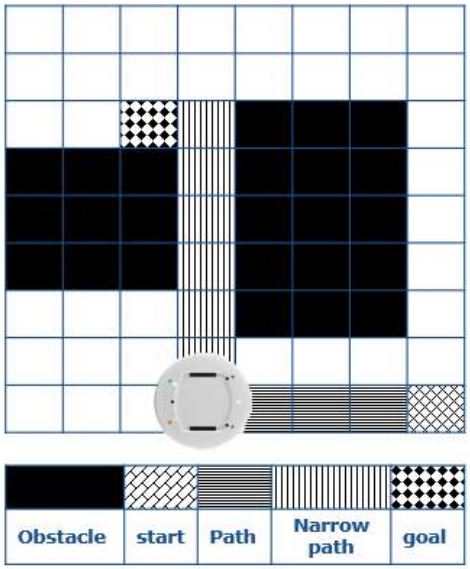


Figure 4.7: Robot size bigger than the cell size.

- If the cell size and the robot size are equal, another problem is emerge. The resistance that one surface or object encounters when moving over another as shown in figure(4.8). Not only the friction between the robot and the obstacles is possible, but also duo to the erroneous distance measurements the robot mistakenly will mark its cell as an obstacle.

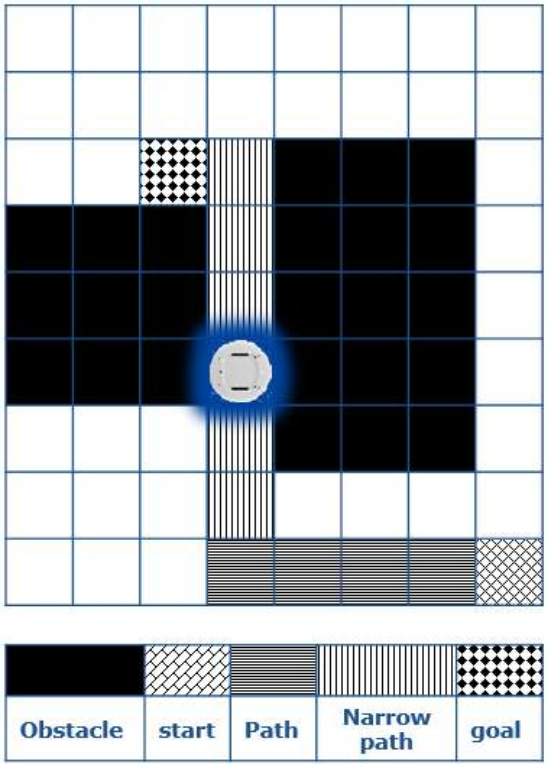


Figure 4.8: Robot size and the cell size are equal.

Finally, for the aforementioned issues we assume that the cell size is equal to size of the robot plus the lowest value of the error range. The cell width and height are presented in figure (4.9) and computed using equations (4.5,4.6).

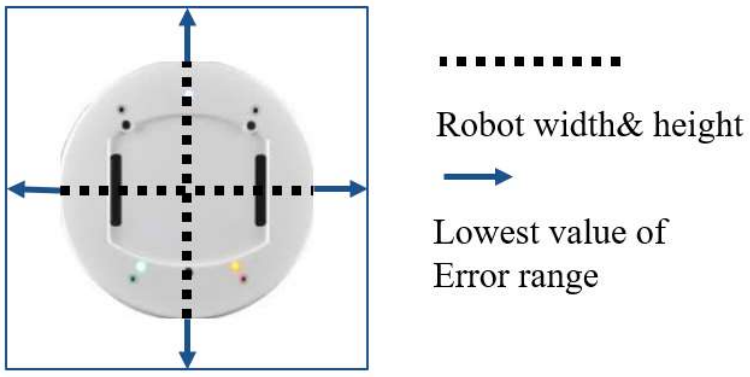


Figure 4.9: Cell size is grater than robot size by ϵ.

$$\mathbf{C}ell\_width = Robot\_width + \epsilon. \tag{4.5}$$

$$\mathbf{C}ell\_height = Robot\_height + \epsilon. \tag{4.6}$$

The path generated by D* based on a global information is divided into segments. Each straight segment in the path is considered as a single path in the APF. The end point of each segment is a sub-goal point for APF. As illustrated in figure (4.10), using APF the robot sub-goal is to reach point (1), once it reaches its goal the new goal becomes point(2) which is an obstacle corner and now APF will guide the robot around the corner . Afterward ,the new goal is point (3) and so on until the robot reaches its final destination.It should be noted that this is a static scenario where there are no dynamic changes in the working environments

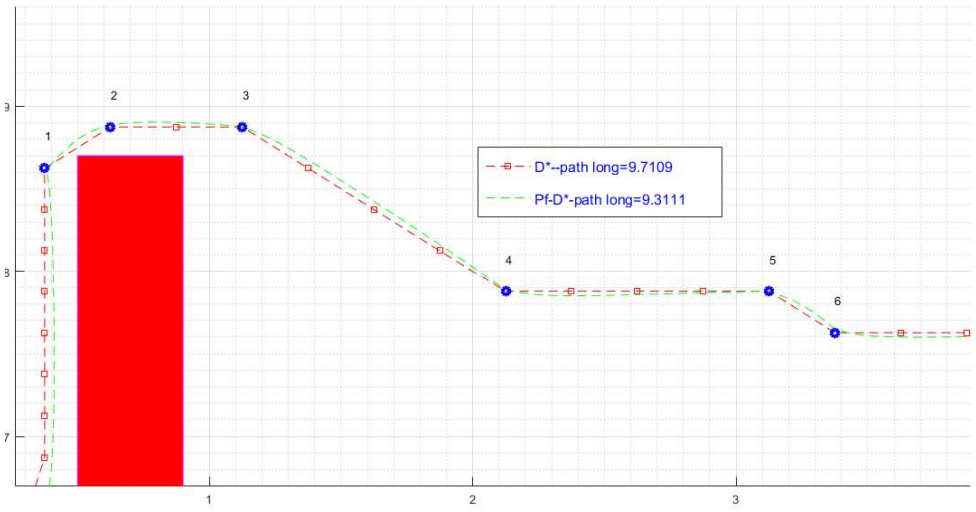


Figure 4.10: Dividing D* path to sub-paths where the end point of each straight segment(in blue) is a goal for the APF in each segment.

A different scenario is presented in figure(4.11) where new obstacles are detected during the recovery process. In this figure , there are four steps that PD* go through until the final path is obtained. The function of these steps are as follows:

1. In step one, the robot begins to follow the initially planned path.

2. The next step starts when the robot discovers an unexpected obstacle . To explain , PD* tries to save the robot from colliding with the detected obstacle and generates a new path.

3. In step three , the robot starts to follow the new path obtained in step two and has no idea about the next unexpected obstacle. Once it discovers the new obstacle it tries to avoid it and generates a new path.

4. Again , a new obstacle is detected and a new path is generated. The robot will follow the final generated path until it reaches its target. However if there exist other obstacles the process will be repeated until the destination is reached.

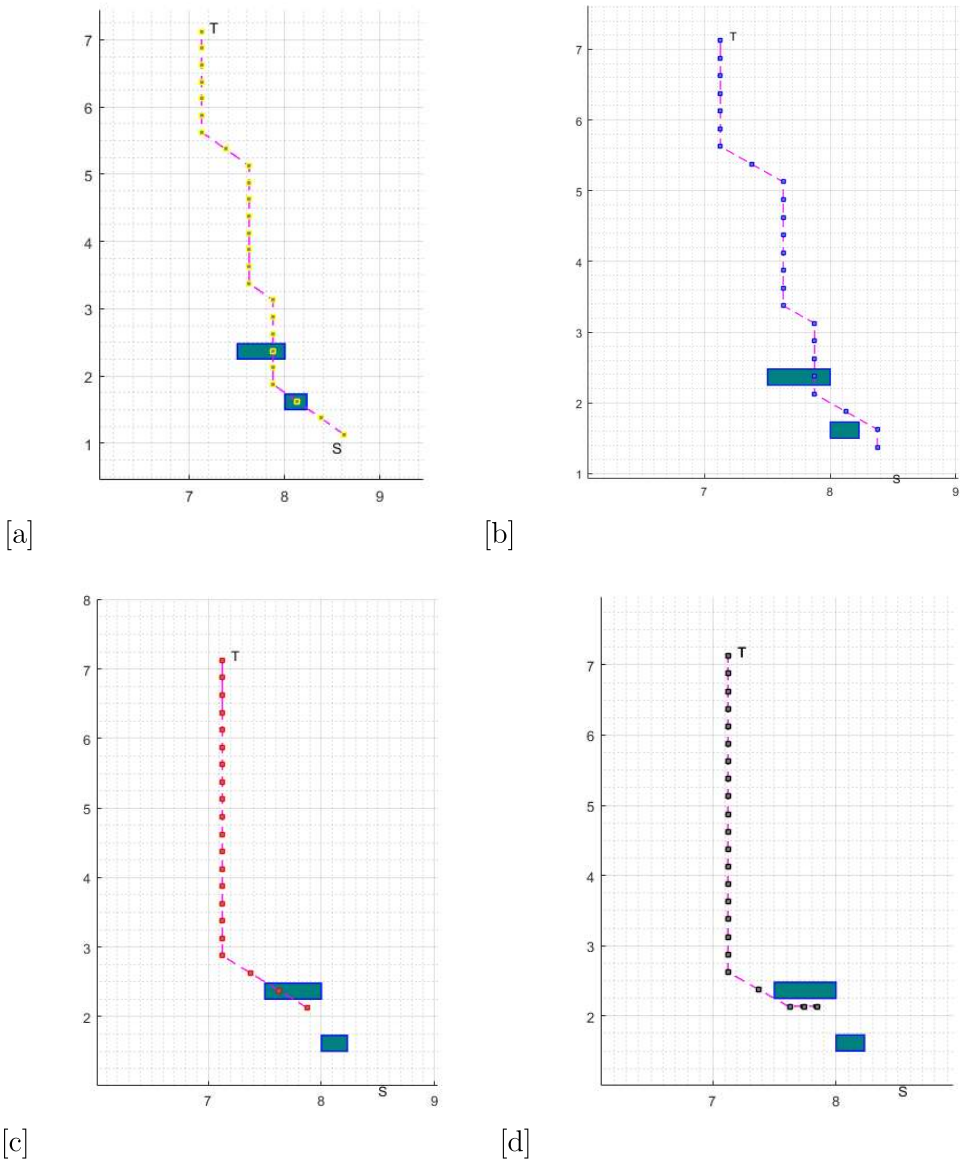
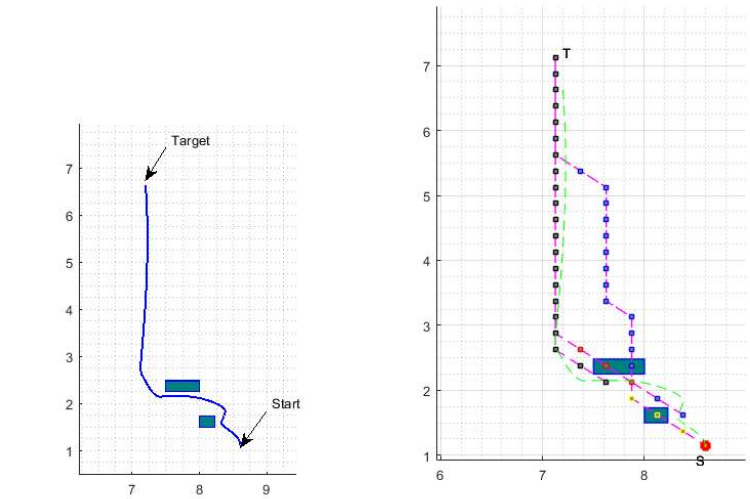


Figure 4.11: PD* step by step.(a) presents the initially planned path,(b) shows how the path change once the first obstacle detected, and (c-d) presents how the path changed when the second obstacle detected.

Figure(4.12,a) demonstrates how safe and smooth the final PD* path is , whereas figure(4.12,b) shows the PD* path along with the four steps explained

earlier . In figure(4.12) (c), D* is compared to our PD* which clearly illustrates the problem of using D* alone in real experiments where change in orientation makes the robot lose its way.



[a] [b]

Figure 4.12: (a)PD* path after the previous four steps, (b) a combination of the four steps with the final path.

4.3 Smooth PD*

In the absence of obstacle, APF gets the shortest path because it is free from cell constraints. While the generated path by a grid-based path planning algorithm like D* and its variants including PD* is not short enough due to the cell restrictions, see figure(4.13).

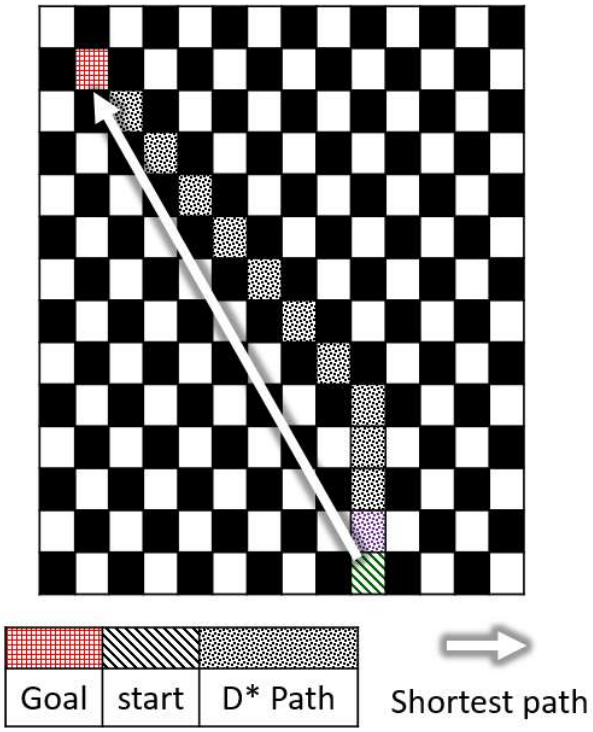


Figure 4.13: D* path versus simple straight path.

In order to optimize PD* path , we proposed a new version of PD* called Smooth PD*. Dijkstra algorithm which is presented in [42], is used to optimize the generated path. Smooth PD* gradually combines two PD* sub-paths at a time into single diagonal one if there is no obstacle between them and update Dijkstra table.

Figure (4.14) shows its steps of finding the shortest path. Incrementally, it starts as in Figure (4.14.a) by combining the first two sub-pathes (in red) and check if there is a known obstacle between them that hinder the robot to move diagonally (in black). If no obstacle is found between the two sub-paths then they are replaced by the diagonal one. After that, a new sub-path is combined with

the diagonal one obtained from the previous step to check whether an obstacle between them is exist or not. The previous steps are repeated until the final path is gained as illustrated in figure(4.14.(b-e)). While, the presence of obstacle either known in advance or unknown makes the diagonal path ineffective. In such case, the first sub-path, that is before the obstacle , is used and the process of smoothing is continue until a complete path is planned, see figure(4.15).

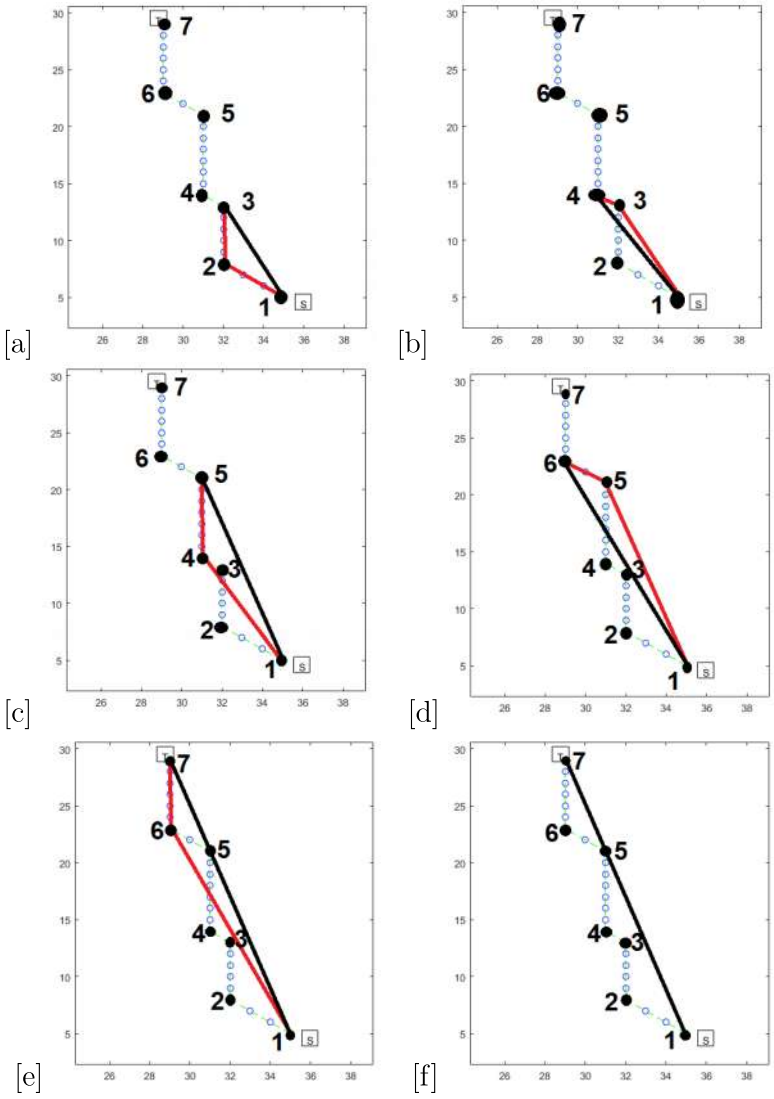


Figure 4.14: Smooth PD* path, (a-e) are the steps of finding the shortest path (in black),(f) is final path.

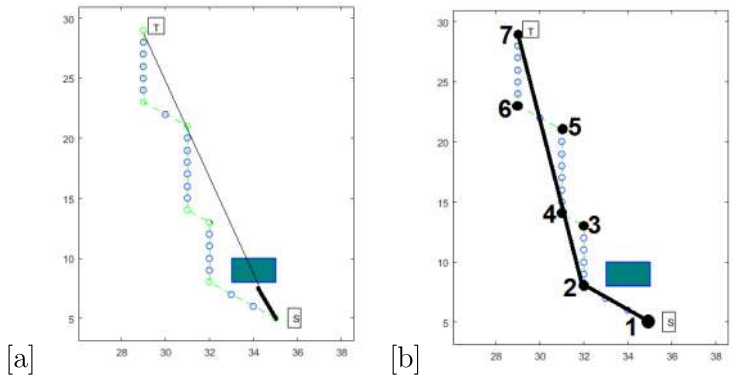
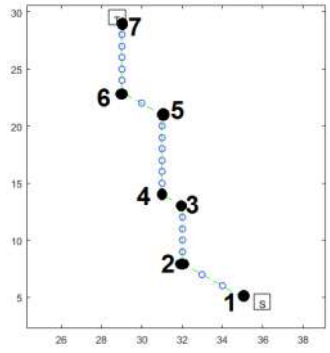


Figure 4.15: Steps of updating the path when an obstacle is detected.

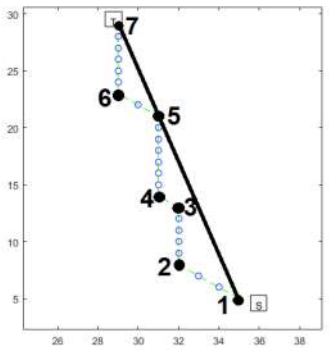
After checking each two segments the Dijkestra table is generated as shown in figure (4.16).



[a]

D(1,7)	1	2	3	4	5	6	7
1		$3\sqrt{2}$	*	*	*	*	*
2	*		5	*	*	*	*
3	*	*		$\sqrt{2}$	*	*	*
4	*	*	*		7	*	*
5	*	*	*	*		$2\sqrt{2}$	*
6	*	*	*	*	*		6
7	*	*	*	*	*	*	

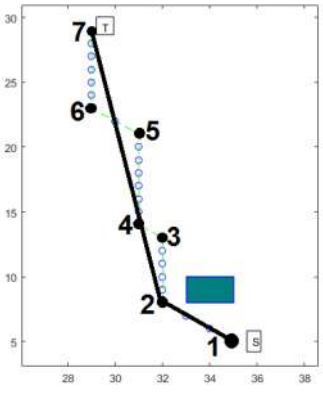
[b]



[c]

D(1,7)	1	2	3	4	5	6	7
1		$3\sqrt{2}$	6.55	6.7	9.68	10	11.6
2	*		5	*	*	*	*
3	*	*		$\sqrt{2}$	*	*	*
4	*	*	*		7	*	*
5	*	*	*	*		$2\sqrt{2}$	*
6	*	*	*	*	*		6
7	*	*	*	*	*	*	

[d]



[e]

D(1,7)	1	2	3	4	5	6	7
1		$3\sqrt{2}$	∞	6.7	9.68	10	11.6
2	*		5	5.1	8.66	9.11	10.90
3	*	*		$\sqrt{2}$	*	*	*
4	*	*	*		7	*	*
5	*	*	*	*		$2\sqrt{2}$	*
6	*	*	*	*	*		6
7	*	*	*	*	*	*	

[f]

Figure 4.16: Dijkestra table. (a-b) the initially planned path and its representation using Dijkestra table,(c-d) is the result of figure(4.14) and its Dijkestra table, (e-f) is the result of figure(4.15) and its generated Dijkestra table.

As illustrated in figure(4.16.d), the shortest path form point 1 to 7 is {1,7}, and the shortest path in figure(4.16.f) is {1,2,7}.

A different scenario using Webots is shown in figure(4.17) , in this scenario

no unknown obstacle is found. The proposed algorithm is much better than all the previous ones. The same scenario is done with new unknown obstacle emerged in the path, see figure(4.18). Once the robot detects a new change in the environments it call D* to replan , and optimize the generated path and utilize APF to fulfill the movement.

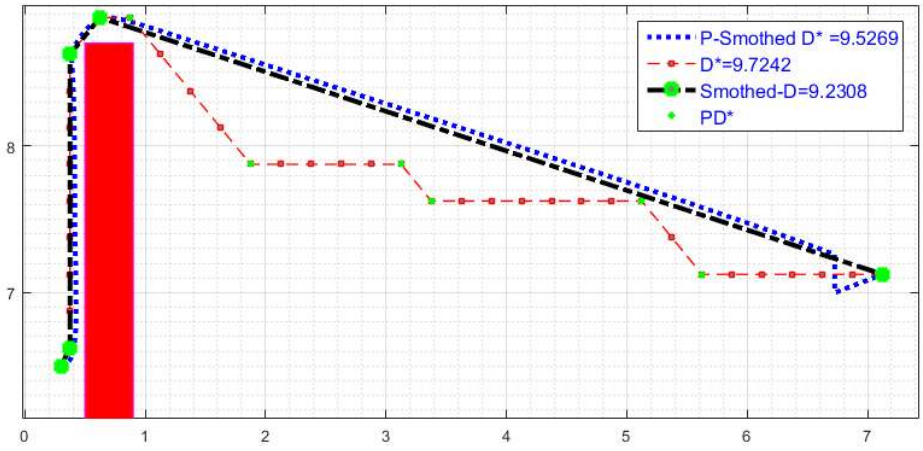


Figure 4.17: Smooth D* versus D*.

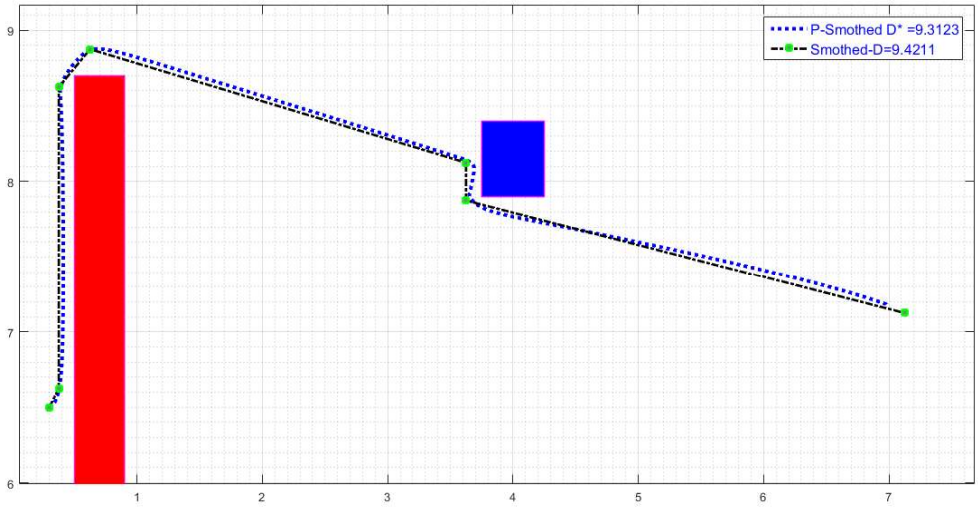


Figure 4.18: A complete example for Smooth D*.

4.4 Secure PD*

The network might get partitioned because of hazardous events that destroy the network agents in the damaged area. Thus sending the recovery team to that area is not a wise decision. Avoiding such area enhances the recovery process and increases the network lifetime by minimizing the number of dead robots. To avoid such areas at all, we need to know how dangerous they are. In order to assess the level of safety of each part in the damaged area there are two ways:

- **Early warning** : when a robot suspects the occurrence of a hazard event it should send a warning broadcast message. This message contains an estimation value that evaluates the level of safety in the working area. This message determines whether the level of safety is high, medium, or low.

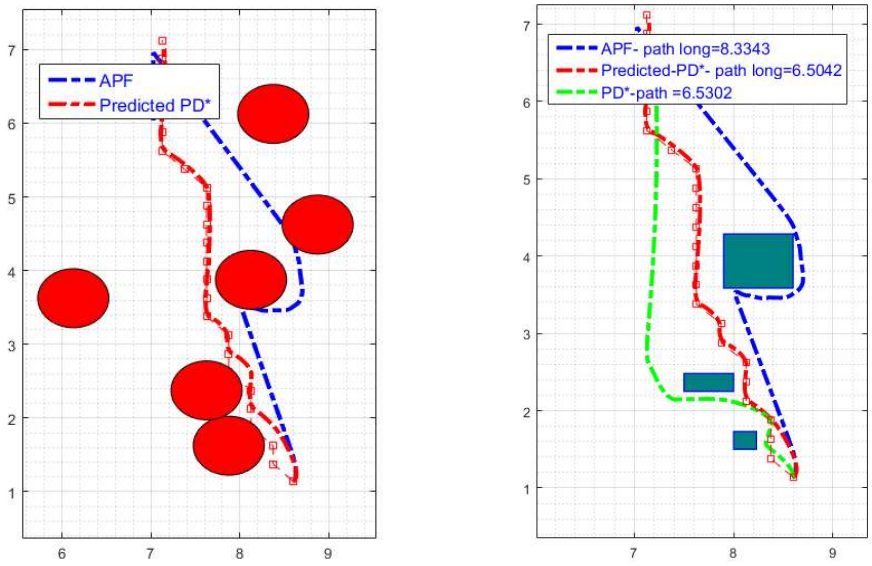
- **Late warning** : once the supervisor starts to acquire the recovery path for the recovery team, it should suspect the damaged area where well-equipped robots have been lost with ambiguous reasons. Such areas are unsafe anymore and a good prediction will help in finding the shortest safe path for the recovery team.

Let the radius of the suspected area is (V_r) which is much less that the communication range(r).

$$V_r << r$$

In the following scenario, figure(4.19), we assume the areas of the missing robots are damaged and we receive a warning message about one of robots and then

make a prediction about the rest. As shown in figure(4.19,b), PD* ends up with a path longer than that of the predicted version.



[a] [b]

Figure 4.19: (a) red circles are the insecure areas , (b) a comparison between PD* and Predicted PD*.

Network restoration recovery path should be collision-free, shortest , and safe. These three path planning objectives are achieved one after another by the partition supervisor and the recovery team leader .Recovery path restoration and the prediction of the secure level of passing through space, are pre-prepared by the supervisor and apply Smoothed PD* to give the leader the secure, shortest, and collision-free path. Then the lader take the responsibility of dynamic path planning during its movement by applying Smoothed PD* if a change occur.

4.5 Dynamic Path Planning Responsibility

Path planning mission is the responsibility of the leader of the recovery team. The leader will use the Predicted Smoothed PD* algorithm. There are several possible ways and for planning the path and minimizing the overall traveled distance. Below are few scenarios.

1. The Leader move and the other node follow: Here, we can imagine two scenarios. The first scenario is where there is no global information available and the actor will act based on just the instantaneous local information. This approach will provide the longest traveled distance. The second scenario is where a prior global information about the area before the damage is known. Then the actor using the instantaneous local information will update the global information and optimize its path. At the same time, it will announce these changes to its followers and so on. This approach is expected to the show an outstanding performance if there is no dead end encountered. We call this solution as survey/adapt/act.

2. The leader scan the area and come back: The main drawback of this approach is that the leader has to traverse the recovery path three times (the first one is for discovering the path, and the others two are to come back and lead the recovery process) which drains the recovery team power and increase the total travel distance by 2L. The total travel distance T_d is com-

puted using equation (4.7).

$$\mathbf{T}_d = \frac{L(\frac{L}{R} - 1)}{2} + 2L \tag{4.7}$$

3. The leader scans the area and leaves pheromone: in this approach, the leader will go first survey the area until it reaches the CoD. While it is traversing the area, it acts like ants where it leaves pheromone when the path is guaranteed. After some time, other nodes will follow using the leader pheromone as a guide. The advantage of this approach is that there is no need for the leader to came back which will minimize the total travel distance by 2L.The total travel distance T_d is computed using equation (4.8). The main shortcomings of this method are that the recovery team need to wait for a sometime that depends on the path length, and the other shortcoming for it is the difficulty in putting the pheromone and how to read it.

$$\mathbf{T}_d = \frac{L(\frac{L}{R} - 1)}{2} \tag{4.8}$$

4. Hybrid from 1st and 2nd scenarios: The Leader move and the other nodes follow until they encounter an obstacle or change in the environment, they stop moving only the leader continue scanning the remaining area and come back. The leader does not have to be a leader after discovering the path. When it comes back to the segment it may just stay still or becomes a follower.

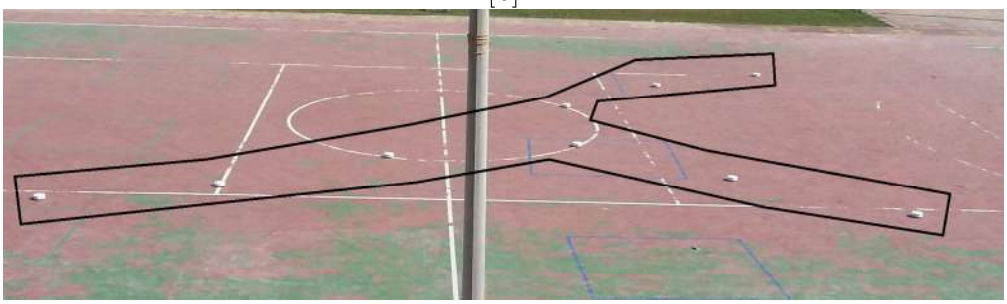
The last scenario is used in this thesis due to its flexibility. The leader uses the proposed multi-objective dynamic path planning method that based on the well-known Focused D* algorithm with an enhancement by utilizing the power of APF and Dijkstra algorithms.

4.6 Integrated real experiment for network restoration and path planning.

The real experiments are conducted using Khepera IV robot. We have only ten robots so we could not test the proposed work with huge number of robots.

[a]

[b]



[c]

Figure 4.20: Real experiment.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

In this work, we have investigated the WSRNs connectivity restoration after a simultaneous robots failure. The proposed solution is based on repositioning the most qualified robots in each partition toward the CoD in a distributed manner. The recovery nodes are selected using fuzzy logic based on their level of power, node rank and the distance. The simulation results have confirmed the effectiveness of our proposed approaches compared to DORMS. Our proposed approaches do not focus mainly in the distance , but they still better than the proposed approaches in the literature in term of total travel distance needed to reestablish the connectivity .Our proposed approach not only federates the partitioned network, but also minimizes the future failure and minimizes the possibility of future

disruption within each partition. Besides selecting the most qualified robots as a recovery team, three deferent recovery path planning approaches were presented by which a shortest, collision-free, and safe path is obtained. As a result of finding the best recovery path, the expected problems that will encounter recovery team is minimized.

5.2 Future Work

As a future work, we are planning to test the proposed approaches in real environment changes with more accurate robots. Also investigate the performance of the proposed approaches under a heterogeneous robots where some of them have the capability of flying, going underwater, or fire fighting . In addition, we are going to enhance CoRFL2 approach to become coverage-aware.

REFERENCES

[1] Ahmad Abbas and Mohamed Younis. Interconnecting disjoint network segments using a mix of stationary and mobile nodes. In *Local Computer Networks (LCN), 2012 IEEE 37th Conference on*, pages 28–35. IEEE, 2012.

[2] KS Al-Sultan and MDS Aliyu. A new potential field-based algorithm for path planning. *Journal of intelligent and robotic systems*, 17(3):265–282, 1996.

[3] Abdullah Alfadhly, Uthman Baroudi, and Mohamed Younis. An effective approach for tolerating simultaneous failures in wireless sensor and actor networks. In *Proceedings of the first ACM international workshop on Mission-oriented wireless sensor networking*, pages 21–26. ACM, 2012.

[4] Hisham M Almasaeid. *Data delivery in fragmented wireless sensor networks using mobile agents*. ProQuest, 2007.

[5] Hisham M Almasaeid and Ahmed E Kamal. Modeling mobility-assisted data collection in wireless sensor networks. In *IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference*, pages 1–5. IEEE, 2008.

[6] Liu Chengqing, Marcelo H Ang, Hariharan Krishnan, and Lim Ser Yong. Virtual obstacle concept for local-minimum-recovery in potential-field based navigation. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 2, pages 983–988. IEEE, 2000.

[7] Benjamin Desjardins, Rafael Falcon, Rami Abielmona, and Emil Petriu. Reliable multiple robot-assisted sensor relocation using multi-objective optimization. In *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pages 4476–4485. IEEE, 2016.

[8] Gregory Dudek and Michael Jenkin. *Computational principles of mobile robotics*. Cambridge university press, 2010.

[9] Zygmunt J Haas, Joseph Y Halpern, and Li Li. Gossip-based ad hoc routing. *IEEE/ACM Transactions on Networking (ToN)*, 14(3):479–491, 2006.

[10] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[11] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[12] Carlos Hernández, Xiaoxun Sun, Sven Koenig, and Pedro Meseguer. Tree adaptive a. In *The 10th International Conference on Autonomous Agents*

and Multiagent Systems-Volume 1, pages 123–130. International Foundation for Autonomous Agents and Multiagent Systems, 2011.

[13] Yong K Hwang and Narendra Ahuja. A potential field approach to path planning. *IEEE Transactions on Robotics and Automation*, 8(1):23–32, 1992.

[14] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The international journal of robotics research*, 5(1):90–98, 1986.

[15] S. Koenig and M. Likhachev. D* lite. 2002.

[16] Sven Koenig, Maxim Likhachev, and David Furcy. Lifelong planning a. *Artificial Intelligence*, 155(1):93–146, 2004.

[17] Jean-Claude Latombe. *Robot motion planning*, volume 124. Springer Science & Business Media, 2012.

[18] Steven M LaValle. Rapidly-exploring random trees: A new tool for path planning. 1998.

[19] Steven M LaValle. *Planning algorithms*. Cambridge university press, 2006.

[20] Sookyoung Lee and Mohamed Younis. Optimized relay placement to federate segments in wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 28(5):742–752, 2010.

[21] Sookyoung Lee and Mohamed Younis. Recovery from multiple simultaneous failures in wireless sensor networks using minimum steiner tree. *Journal of Parallel and Distributed Computing*, 70(5):525–536, 2010.

[22] Maxim Likhachev and Sven Koenig. A generalized framework for lifelong planning a* search. In *ICAPS*, pages 99–108, 2005.

[23] GC Luh and WW Liu. Motion planning for mobile robots in dynamic environments using a potential field immune network. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 221(7):1033–1045, 2007.

[24] V Lumelsky and A Stepanov. Dynamic path planning for a mobile automaton with limited information on the environment. *IEEE transactions on Automatic control*, 31(11):1058–1063, 1986.

[25] Na Lv and Zuren Feng. Numerical potential field and ant colony optimization based path planning in dynamic environment. In *2006 6th World Congress on Intelligent Control and Automation*, volume 2, pages 8966–8970. IEEE, 2006.

[26] Gerard T McKee and Paul S Schenker. Networked robotics. In *Intelligent Systems and Smart Manufacturing*, pages 197–209. International Society for Optics and Photonics, 2000.

[27] Achille Melingui, Taha Chettibi, Rochdi Merzouki, and Jean Bosco Mbede. Adaptive navigation of an omni-drive autonomous mobile robot in unstructured dynamic environments. In *Robotics and Biomimetics (ROBIO), 2013 IEEE International Conference on*, pages 1924–1929. IEEE, 2013.

[28] G Ayorkor Mills-Tettey, Anthony Stentz, and M Bernardine Dias. Dd* lite: Efficient incremental search with state dominance. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, page 1032. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.

[29] Aminu Mohammed, Mohamed Ould-Khaoua, Lewis M Mackenzie, Colin Perkins, and Jamal-Deen Abdulai. Probabilistic counter-based route discovery for mobile ad hoc networks. In *Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly*, pages 1335–1339. ACM, 2009.

[30] Alex Nash, Kenny Daniel, Sven Koenig, and Ariel Felner. Theta^*: Any-angle path planning on grids. In *Proceedings of the national conference on artificial intelligence*, volume 22, page 1177. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.

[31] Alex Nash, Sven Koenig, and Maxim Likhachev. Incremental phi*: Incremental any-angle path planning on grids. 2009.

[32] Alex Nash, Sven Koenig, and Craig Tovey. Lazy theta*: Any-angle path planning and path length analysis in 3d. In *Third Annual Symposium on Combinatorial Search*, 2010.

[33] Subbash Panati, Bayanjargal Baasandorj, and Kil To Chong. Autonomous mobile robot navigation using harmonic potential field. In *IOP Conference*

Series: Materials Science and Engineering, volume 83, page 012018. IOP Publishing, 2015.

[34] Roland Philippsen. *Motion planning and obstacle avoidance for mobile robots in highly cluttered dynamic environments*. PhD thesis, ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE, 2004.

[35] Virender Ranga, Mayank Dave, and Anil Kumar Verma. Network partitioning recovery mechanisms in wsans: a survey. *Wireless personal communications*, 72(2):857–917, 2013.

[36] Timothy J Ross. *Fuzzy logic with engineering applications*. John Wiley & Sons, 2009.

[37] Fatih Senel and Mohamed Younis. Optimized connectivity restoration in a partitioned wireless sensor network. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pages 1–5. IEEE, 2011.

[38] Fatih Senel and Mohamed Younis. Relay node placement in structurally damaged wireless sensor networks via triangular steiner tree approximation. *Computer Communications*, 34(16):1932–1941, 2011.

[39] Fatih Senel and Mohamed Younis. Optimized interconnection of disjoint wireless sensor network segments using k mobile data collectors. In *ICC*, pages 492–496, 2012.

[40] Fatih Senel and Mohamed Younis. Optimized relay node placement for establishing connectivity in sensor networks. In *Global Communications Conference (GLOBECOM), 2012 IEEE*, pages 512–517. IEEE, 2012.

[41] Fatih Senel, Mohamed F Younis, and Kemal Akkaya. Bio-inspired relay node placement heuristics for repairing damaged wireless sensor networks. *IEEE Transactions on Vehicular Technology*, 60(4):1835–1848, 2011.

[42] S Skiena. Dijkstras algorithm. *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica, Reading, MA: Addison-Wesley*, pages 225–227, 1990.

[43] Jerome LVM Stanislaus and Mohamed Younis. Delay-conscious federation of multiple wireless sensor network segments using mobile relays. In *Vehicular Technology Conference (VTC Fall), 2012 IEEE*, pages 1–5. IEEE, 2012.

[44] Jerome LVM Stanislaus and Mohamed Younis. Mobile relays based federation of multiple wireless sensor network segments with reduced-latency. In *2013 IEEE International Conference on Communications (ICC)*, pages 6407–6411. IEEE, 2013.

[45] Anthony Stentz. Optimal and efficient path planning for partially-known environments. In *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pages 3310–3317. IEEE, 1994.

[46] Anthony Stentz et al. The focussed dˆ* algorithm for real-time replanning. In *IJCAI*, volume 95, pages 1652–1659, 1995.

[47] Xiaoxun Sun, Sven Koenig, and William Yeoh. Generalized adaptive a*. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '08, pages 469–476, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems.

[48] Xiaoxun Sun, William Yeoh, and Sven Koenig. Dynamic fringe-saving a. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 891–898. International Foundation for Autonomous Agents and Multiagent Systems, 2009.

[49] Prahlad Vadakkepat, Tong Heng Lee, and Liu Xin. Application of evolutionary artificial potential field in robot soccer system. In *IFSA World Congress and 20th NAFIPS International Conference, 2001. Joint 9th*, pages 2781–2785. IEEE, 2001.

[50] Lim Chee Wang, Lim Ser Yong, and Marcelo H Ang. Hybrid of global path planning and local navigation implemented on a mobile robot in indoor environment. In *Intelligent Control, 2002. Proceedings of the 2002 IEEE International Symposium on*, pages 821–826. IEEE, 2002.

[51] Andrew Wichmann, Burcu Demirelli Okkalioglu, and Turgay Korkmaz. The integration of mobile (tele) robotics and wireless sensor networks: A survey. *Computer Communications*, 51:21–35, 2014.

[52] Brian H Wilcox, Todd Litwin, Jeff Biesiadecki, Jaret Matthews, Matt Heverly, Jack Morrison, Julie Townsend, Norman Ahmad, Allen Sirota, and Brian Cooper. Athlete: A cargo handling and manipulation robot for the moon. *Journal of Field Robotics*, 24(5):421–434, 2007.

[53] Chunhui Wu, Hongsheng Chen, and Jiangang Liu. A survey of connectivity restoration in wireless sensor networks. In *Consumer Electronics, Communications and Networks (CECNet), 2013 3rd International Conference on*, pages 65–67. IEEE, 2013.

[54] Mohamed Younis, Izzet F Senturk, Kemal Akkaya, Sookyoung Lee, and Fatih Senel. Topology management techniques for tolerating node failures in wireless sensor networks: A survey. *Computer Networks*, 58:254–283, 2014.

Vitae

- Name: Mohammed Yahya Al-Darwbi
- Nationality: Yemen
- Date of Birth: 1/1/1984
- Email: *m.aldarowbi@gmail.com*
- Permenant Address: Yemen, Thamar, Heran
- Publication:

  [1] Baroudi Uthman, and Mohammed Aldarwbi. "CoRFL: A Connectivity Restoration Mechanism Using Fuzzy Logic in Wireless Actor and Sensor Networks." Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on. IEEE, 2015.

  [2] Mohammed Aldarwbi, and Baroudi Uthman. E-SRFL: Energy-aware Simultaneous Res Restoration mechanism Using Fuzzy Logic in Wireless Sensor and Robotics Networks Journal paper.(ongoing)

- Submitted patents:

[1] Mohammed Aldarwbi, and Baroudi Uthman. CoRFL: A Connectivity Restoration Mechanism Using Fuzzy Logic in Wireless Actor and Sensor Networks.